MOCKING BOARD

技術應用

麥法基著



電腦時代出版社

序

自從 MOCKINGBOARD 被介紹到萍果電腦以來，無論聲音效果或電腦音樂領域都大大地擴展了。事實上，以售價低於三佰元一張聲音擴展附件而言，所獲得的價值亦並非金錢所能衡量。不但對電腦編寫程式方面可以達到更佳效果，而且對於學習音樂方面更極有幫助，這個專輯便是針對以上兩點而編寫的。

筆者承認現時 MOCKINGBOARD 軟件仍然處於開始階段，一方面由於該附件在外國未完全通行，另一方面就編寫程式而言，設計者需對機械語言有較深認識，造成一般 BASIC 設計者卻步不前。

既然電腦已經是一件普及的家庭用品而不是一副神秘的機器，我們絕不能讓一件有價值的附件只是遊戲的一部份，所以我們編寫一個專輯對 MOCKINGBOARD 作出幾個深入淺出的討論，希望由此引出各位讀者更寶貴的意見和經驗。

本專輯以一般 BASIC 作爲主要程式語言，當然亦需要機械語言作爲媒介。有需要時，筆者會提議讀者用 COMPILER 來獲得較佳效果，這樣對一般不熟識機械語的朋友亦能容易掌握程式運行。

以下是本專輯所應用的軟件：

MERLIN ASSEMBLER
TASC COMPILER
PIXIT GRAPHIC PROCESSOR
GRAPHICS MAGICIAN
MOCKINGBOARD SOUND UTILITIES
MUSIC CONSTRUCTION SET

第一章

MOCKING BOARD 和聲音關係

MOCKINGBOARD 一共有6個聲道產生聲音，分別由左右兩邊喇叭輸送出來，因此可以使我們獲得美妙的立體聲效果。究竟這些聲音從何而來呢？

要令 MOCKINGBOARD 產生聲音，我們必須供給它一些所需資料， MOCKINGBOARD 本身是電腦的附件，它並非記憶體的一部份，所以我們不能好像存取記憶的資料一樣把資料輸送，必須通過幾個過程使它產生聲音，而這些過程當然是以機械語言來達成任務，因爲所有和電腦外界溝通都需要極快的速度。

參看圖1.1，這裡有十四個不同記存器（REGISTER）

FIG. 1.1

Register Name	Address	Symbol	Name
0	$C400	PBD	Input/Output port B
1	$C401	PAD	Input/Output port A
2	$C402	DDRB	Port B data direction register
3	$C403	DDRA	Port A data direction register
4	$C404	T1CL	Latch low-order counter
5	$C405	T1CH	Latch high-order counter
6	$C406	T1LL	Low-order latch
7	$C407	T1LH	High-order latch
8	$C408	T2CL	Latch low order counter2
9	$C409	T2CH	Latch high order counter2
A	$C40A	SR	Shift register
B	$C40B	ACR	Auxiliary control rigister
C	$C40C	PCR	Peripheral control register
D	$C40D	IFR	Interupt flag register
E	$C40E	IER	Interupt enable register
F	$C40F	PAD	Same as 1 without handshake

負責十四種不同資料，倘若這資料完全被輸入所有記存器內，則聲音立即產生出來。很明顯這些記存器並非記憶位址，要達到輸送目的，便要倚靠幾個副程式來進行。下面就是幾個副程式的作用：

第一種副程式稱之爲 PRIMARY ROUTINES ，基本副程式。無論那一個時候要產生聲音，我們必須要呼叫其中之一個，或數個。

INIT 開啟 MOCKINGBOARD 程式

讓我們看看 INIT 開啓程式，當 APPLE 被接上電源或按 RESET , MOCKINGBOARD 都會被視爲輸入附件。通常 MOCKINGBOARD 都會被安放在第4 SLOT 上，所以＄C400至＄C403是第一個喇叭的輸出/輸入部份，而＄C480至＄C483則是第二個喇叭的輸出/輸入部份。我們首先研究第一個喇叭，該位址分別代表下面幾個名稱：

＄C400	ORB
＄C401	ORA
＄C402	DDRB
＄C403	DDRA

在未有任何資料輸入之前， MOCKINGBOARD 存在一些未解決的問題，因爲 MOCKINGBOARD 是用6522的晶片作爲溝通媒介，這枚晶片並不一定用來產生聲音，它可以做任何控制其他機器之用，例如印字機、鍵盤、時間控制、溫度控制等，這些機器有些是輸入作用，有些是輸出作用，亦可以是輸入和輸出的組合，於是問題便產生了，究竟這枚晶片現在是輸入還是輸出？控制些什麼類型機器？怎樣輸入？怎樣輸出？

```
:ASM
                    1   * PRIMARY ROUTINES
                    2   * FOR SLOT 4
                    3   *
                    4             ORG   $9000
                    5   * ADDRESSES FOR FIRST 6522
                    6   ORB       EQU   $C400      ;PORT B
                    7   ORA       EQU   $C401      ;PORT A
                    8   DDRB      EQU   $C402      ;DATA DIRECTION REGISTER (B)
                    9   DDRA      EQU   $C403      ;DATA DIRECTION REGISTER (A)
                    10  * ADDRESS FOR SECOND 6522
                    11  ORB2      EQU   $C480      ;PORT B
                    12  ORA2      EQU   $C481      ;PORT A
                    13  DDRB2     EQU   $C482      ;DATA DIRECTION REGISTER (B)
                    14  DDRA2     EQU   $C483      ;DATA DIRECTION REGISTER (A)
                    15  *
                    16  * ROUTINES FOR FIRST 6522
                    17  *
9000: A9 FF         18  INIT      LDA   #$FF       ;SET PORT A FOR OUTPUT
9002: 8D 03 C4      19            STA   DDRA
9005: A9 07         20            LDA   #$07       ;SET PORT B FOR OUTPUT
9007: 8D 02 C4      21            STA   DDRB
900A: 60            22            RTS
                    23  *
900B: A9 07         24  LATCH     LDA   #$07       ;SEND "LATCH COMMAND"
900D: 8D 00 C4      25            STA   ORB        ;TO SOUND CHIP
9010: A9 04         26            LDA   #$04       ;THROUGH PORT B
9012: 8D 00 C4      27            STA   ORB
9015: 60            28            RTS              ;RETURN
                    29  *
9016: A9 06         30  WRITE     LDA   #$06       ;SEND "WRITE COMMAND"
9018: 8D 00 C4      31            STA   ORB        ;TO SOUND CHIP
901B: A9 04         32            LDA   #$04       ;THROUGH PORT B
901D: 8D 00 C4      33            STA   ORB
9020: 60            34            RTS              ;RETURN
                    35  *
9021: A9 00         36  RESET     LDA   #$00       ;SEND "RESET COMMAND"
9023: 8D 00 C4      37            STA   ORB        ;TO SOUND CHIP
9026: A9 04         38            LDA   #$04       ;THROUGH PORT B
9028: 8D 00 C4      39            STA   ORB
902B: 60            40            RTS              ;RETURN
                    41  *
                    42  * ROUTINES FOR SECOND 6522
                    43  *
902C: A9 FF         44  INIT2     LDA   #$FF       ;SET PORT A FOR OUTPUT
902E: 8D 83 C4      45            STA   DDRA2
9031: A9 07         46            LDA   #$07       ;SET PORT B FOR OUTPUT
9033: 8D 82 C4      47            STA   DDRB2
9036: 60            48            RTS
                    49  *
9037: A9 07         50  LATCH2    LDA   #$07       ;SEND "LATCH COMMAND"
9039: 8D 80 C4      51            STA   ORB2       ;TO SOUND CHIP
903C: A9 04         52            LDA   #$04       ;THROUGH PORT B
903E: 8D 80 C4      53            STA   ORB2
9041: 60            54            RTS              ;RETURN
                    55  *
9042: A9 06         56  WRITE2    LDA   #$06       ;SEND WRITE COMMAND
9044: 8D 80 C4      57            STA   ORB2       ;TO SOUND CHIP
9047: A9 04         58            LDA   #$04       ;THROUGH PORT B
9049: 8D 80 C4      59            STA   ORB2
904C: 60            60            RTS              ;RETURN
                    61  *
904D: A9 00         62  RESET2    LDA   #$00       ;SEND "RESET COMMAND"
904F: 8D 80 C4      63            STA   ORB2       ;TO SOUND CHIP
9052: A9 04         64            LDA   #$04       ;THROUGH PORT B
9054: 8D 80 C4      65            STA   ORB2
9057: 60            66            RTS              ;RETURN

--End assembly--

88 bytes

Errors: 0

Symbol table - alphabetical order:
```

在這裡，所有資料都會按次序影响 ORB，這裡是 MOCKINGBOARD 工作的地方，開啓程式首先在 DDRA（＄C403）存入＄FF，這個數值是＄00的話，則 MOCKINGBOARD 會作爲輸入，不會產生聲音，如果是＄FF 的話，則整個 MOCKINGBOARD 會轉爲輸出，會接受電腦供給的資料。

雖然屬於輸出狀態，但究竟要處理那一類型機器呢？這是 DDRB 的數值負責，＄07是喇叭數值，假如是印字機的話，則視乎那一類型印字機而定，有些印字機是＄37，不同的機器有不同數值。一些複雜的方程式計算機器的脈冲時間，則不在此討論範圍。

經過呼叫這個 INIT 副程式之後，則第一喇叭被開啓，所有記存器會開始準備接受資料了，同理 INIT2 是開啓第二喇叭之用。

LATCH 設定記存器工作副程式

十四個記存器既然被開啓可以接受資料，但這些資料做些甚麼工作呢？現時 MOCKINGBOARD 不能分辨究竟資料是甚麼東西。LATCH 是告訴 MOCKINGBOARD 把 PORT B 作爲位址之用。0記存器的位址是0，1記存器的位址是1。這時倘若我把零這個數值存入 PORT A，呼叫 LATCH 則 MOCKINGBOARD 會知道你應用第0號記存器。LDA ＃＄07是把輸出 PORT B 暫作記存器，由 PORT A 數值而決定那一種記存器。至於再存入＄04的數值是 PORT B 工作的方法和脈冲時間，這是硬體所需數值，亦不在此討論。

這裡相信你找不到 PORT A 的數值，因爲 LATCH 是一個基本程式，並非立即可以應用，有別於 INIT 程式。事先必須由另外一個程式設定 PORT A 的數值，而且不止一個記存器位址，這是由 TABLE ACCESS ROUTINE 所負責，下面會談到。

WRITE 輸入資料副程式

記存器性質既然被設定，則可以輸入資料了。資料是產生什麼類型聲音的數值表，這時把資料再輸入 PORT A，呼叫 WRITE 則 ORB 便會把資料輸入，LDA ＃＄06是把 ORB 由記存器性質轉爲輸入聲音資料，同樣地 PORT A 數值由 TABLE ACCESS ROUTINE 所決定。

RESET 清除記存器數值副程式

聲音一經產生，它會永遠存在 MOCKINGBOARD 內，倘若要變爲另外一種聲音，事先必須清除 BOARD 內資料，讓新的數值輸入，否則兩種聲音混而爲一。RESET 便是把舊有資料清除。不過 RESET 會立即把十四個記存器資料全部消去，如果某些情況下要保留有聲音而加上新的聲音，則不能應用此程式。

TABLE ACCESS聲音表副程式

這個副程式負責推動 MOCKINGBOARD 產生聲音，分別處理兩種工作過程。在程式第6行的 PTR 是聲音表指位器，第7行是 PORT A 位址，這裡相信你可以發覺 Y-

REGISTER 是用來指示記存器的號碼，同樣亦是聲音表那一個資料，因爲記存器號碼和資料次序是順次序相同的。程式第2行首先輸入 Y-REGISTER 數值到 PORT A，然後呼叫 LATCH 設定記存器號碼，跟着第24行找出聲音表數值，再存入 PORT A，然後呼叫 WRITE 把資料存入記存器內，如是者直至所有十四個資料輸送完畢，則聲音便產生了。

PORT A 要被運用兩次的原因是 LATCH 和 WRITE 都要利用它作爲決定 PORT B 性質之用。而記存器和聲音資料都依着次序排列，所以可以共用第21行的 Y-REGISTER。

聰明的讀者也許明白到並不是一定要把全部十四個記存器資料輸入。有時很多特殊聲音效果可以一方面有一種聲音產生作爲背景，而一些聲音亦可以隨時產生，視乎如何靈活運用 LATCH 和 WRITE 副程式而已。SKY FOX 就是一個很好的例子。

```
                    1    * TABLE ACCESS ROUTINE
                    2    * FOR SLOT 4
                    3    *
                    4              ORG  $8000
                    5    * ADDRESSES FOR FIRST 6522
                    6    PTR       EQU  $08          ;DATA POINTER
                    7    ORA       EQU  $C401        ;PORT A
                    8    LATCH     EQU  $900B        ;LATCH SUBROUTINE
                    9    WRITE     EQU  $9016        ;WRITE SUBROUTINE
                    10   RESET     EQU  $9021        ;RESET SUBROUTINE
                    11   * ADDRESS FOR SECOND 6522
                    12   PTR2      EQU  $0A          ;DATA POINTER
                    13   ORA2      EQU  $C481        ;PORT A
                    14   LATCH2    EQU  $9037        ;LATCH SUBROUTINE
                    15   WRITE2    EQU  $9042        ;WRITE SUBROUTINE
                    16   RESET2    EQU  $904D        ;RESET SUBROUTINE
                    17   *
                    18   * ROUTINES FOR FIRST 6522
                    19   *
8000: 20 21 90      20   START     JSR  RESET        ;RESET SOUND CHIP
8003: A0 00         21             LDY  #$00         ;USED TO IDENTIFY REGISTER
8005: 8C 01 C4      22   LOOP      STY  ORA          ;# OF SOUND CHIP
8008: 20 0B 90      23             JSR  LATCH
800B: B1 08         24             LDA  (PTR),Y      ;GET DATA FROM TABLE
800D: 8D 01 C4      25             STA  ORA
8010: 20 16 90      26             JSR  WRITE        ;STORE IN REGISTER
8013: C0 0F         27             CPY  #$0F         ;END OF DATA?
8015: F0 04         28             BEQ  DONE         ;YES, EXIT
8017: C8            29             INY
8018: 4C 05 80      30             JMP  LOOP         ;NO, GET NEXT SET
801B: 60            31   DONE      RTS               ;RETURN
                    32   *
                    33   * ROUTINES FOR SEND 6522
                    34   *
801C: 20 4D 90      35   START2    JSR  RESET2
801F: A0 00         36             LDY  #$00
8021: 8C 81 C4      37   LOOP2     STY  ORA2
8024: 20 37 90      38             JSR  LATCH2
8027: B1 0A         39             LDA  (PTR2),Y
8029: 8D 81 C4      40             STA  ORA2
802C: 20 42 90      41             JSR  WRITE2
802F: C0 0F         42             CPY  #$0F
8031: F0 04         43             BEQ  DONE2
8033: C8            44             INY
8034: 4C 21 80      45             JMP  LOOP2
8037: 60            46   DONE2     RTS

--End assembly--
```

第二章

MOCKINGBOARD 基本運用

在 MOCKINGBOARD 手冊內，有一個 SOUND UTILITY，相信大家都會明白如何應用。筆者在第13期的電腦時代亦曾提及關於 SOUND UTILITY 各種調節，這裡我們再深入研究 MOCKINGBOARD 的基本運用。

上一章我們知道有一個 TABLE ACCESS 副程式負責把資料輸入記存器內讓 MOCKINGBOARD 產生聲音，TABLE 當然指把資料造成一個列表，讓電腦由表內查閱各種不同資料，順序輸入 PORT A。查表工作完畢，聲音自然產生了。不過我們總不能亂打亂撞去供應資料，況且每種程式對聲音效果有不同要求。而 SOUND UTILITY 便是讓我們事先把資料編製及測試妥當，最後把列表編制存於電腦指定記憶內，使 TABLE ACCESS 有所根據。

參看圖2.1，這是十四個記存器（REGISTER）的工作內容。順次序由 R0 至 R13，現簡述如下：

CHANNEL 聲道，每一邊喇叭可以同時產生三個不同聲音，分三個聲道輸出。而每一個聲道可以是純樂音，或噪音，甚至兩者混合。

TONE PERIOD 音域，這是指純樂音而言，每一個聲道有兩種不同的調節，分為細緻（FINE TUNE）及粗略（COARSE TUNE）兩種，用以調節音域的高低。

圖2.1

	REGISTER	DESCRIPTION	DEC	HEX
R0	CHANNEL A TONE PERIOD	FINE TUNE	0-255	00-FF
R1		COARSE TUNE	0-15	00-0F
R2	CHANNEL B TONE PERIOD	FINE TUNE	0-255	00-FF
R3		COARSE TUNE	0-15	00-0F
R4	CHANNEL C TONE PERIOD	FINE TUNE	0-255	00-FF
R5		COARSE TUNE	0-15	00-0F
R6	NOISE PERIOD	ALL CHANNELS	0-31	00-1F
R7	ENABLE	NOISE/TONE	0-63	00-3F
R8	CHANNEL A AMPLITUDE	AMPLITUDE LEVEL MODE SELECT FIXED=0-15 VARIABLE=16	0-16	00-10
R9	CHANNEL B AMPLITUDE		0-16	00-10
R10	CHANNEL C AMPLITUDE		0-16	00-10
R11		FINE TUNE ENVELOPE	0-255	00-FF
R12	ENVELOPE PERIOD	COARSE TUNE ENVELOPE	0-255	00-FF
R13	ENVELOPE SHAPE/CYCLE	CONT:ATTACK: ALT:HOLD	0-15	00-0F
R14	NOT USED	REGISTER VALUE NOT SIGNIFICANT		
R15	NOT USED	REGISTER VALUE NOT SIGNIFICANT		

NOISE PERIOD 噪音，聲音除了純樂音之外，更包括噪音在內，這樣可以產生自然界的聲音。

ENABLE 純音和噪音的選擇。每一個聲道都可包括全部純音，純音和噪音混合，或全部是噪音。

AMPLITUDE 音量，雖然 MOCKINGBOARD 上可以調節音量，但這是指整個喇叭的音量控制。AMPLITUDE 是個別聲道的音量調節，而且還加上一個 VARIABLE，該數值是16，使聲音根據不同需要變化。

ENVELOPE PERIOD 和 ENVELOPE SHAPE

音形調節和音形。每一種聲音可以給予一種特殊花樣或方式讓它跟着產生然後重複。這些花樣我們姑且稱之爲音形，MOCKINGBOARD 總共有八種（參看圖2.2），而音形調節便是把這些形狀加以壓縮或擴張，造成不同的效果。

聲音效果的測試

下面我們更加詳細地把以上記存器工作實際體驗。請預備一張已經有 DOS 3.3的磁碟，然後鍵入 CALL-151進入監督程式，再把下面一小段程式鍵入，以 TABLE ACCESS 的名稱 BSAVE 到磁碟上，我們要利用它來測試各種聲音效果。

```
8F00- A5 FC 8D 01 C4 20 0B 90
8F08- A5 FE 8D 01 C4 20 16 90
8F10- 60 A5 FD 8D 81 C4 20 37
8F18- 90 A5 FF 8D 81 C4 20 16
8F20- 90 60
```

圖2.2

DEC	HEX	CONT	ATTK	ALT	HOLD	GRAPHIC REPRESENTATION
08	08	ON	--	--	--	
09	09	ON	--	--	ON	
10	0A	ON	--	ON	--	
11	0B	ON	--	ON	ON	
12	0C	ON	ON	--	--	
13	0D	ON	ON	--	ON	
14	0E	ON	ON	ON	--	
15	0F	ON	ON	ON	ON	

SHAPE PATTERNS 0-7 ARE ONE CYCLE PATTERNS AND MAY BE REPRESENTEDS ABOVE.FOR EXAMPLE,SHAPE PATTERN 0 IS A DUPLICATE OF PATTEN 9.
A DUPLICATE OF PATTERN 9.

DEC	HEX	CONT	ATTK	ALT	HOLD	GRAPHIC REPRESENTATION
00	00	--	--	X	X	

圖 2.3

REGISTER VALUE		NOISE CHANNEL			TONE CHANNEL			REGISTER VALUE		NOISE CHANNEL			TONE CHANNEL		
DEC	HEX	C	B	A	C	B	A	DEC	HEX	C	B	A	C	B	A
00	00	ON	ON	ON	ON	ON	ON	32	20	--	ON	ON	ON	ON	ON
01	01	ON	ON	ON	ON	ON	--	33	21	--	ON	ON	ON	ON	--
02	02	ON	ON	ON	ON	--	ON	34	21	--	ON	ON	ON	--	ON
03	03	ON	ON	ON	ON	--	--	35	23	--	ON	ON	ON	--	--
04	04	ON	ON	ON	--	ON	ON	36	24	--	ON	ON	--	ON	ON
05	05	ON	ON	ON	--	ON	--	37	25	--	ON	ON	--	ON	--
06	06	ON	ON	ON	--	--	ON	38	26	--	ON	ON	--	--	ON
07	07	ON	ON	ON	--	--	--	39	27	--	ON	ON	--	--	--
08	08	ON	ON	--	ON	ON	ON	40	28	--	ON	--	ON	ON	ON
09	09	ON	ON	--	ON	ON	--	41	29	--	ON	--	ON	ON	--
10	0A	ON	ON	--	ON	--	ON	42	2A	--	ON	--	ON	--	ON
11	0B	ON	ON	--	ON	--	--	43	2B	--	ON	--	ON	--	--
12	0C	ON	ON	--	--	ON	ON	44	2C	--	ON	--	--	ON	ON
13	0D	ON	ON	--	--	ON	--	45	2D	--	ON	--	--	ON	--
14	0E	ON	ON	--	--	--	ON	46	2E	--	ON	--	--	--	ON
15	0F	ON	ON	--	--	--	--	47	2F	--	ON	--	--	--	--
16	10	ON	--	ON	ON	ON	ON	48	30	--	--	ON	ON	ON	ON
17	11	ON	--	ON	ON	ON	--	49	31	--	--	ON	ON	ON	--
18	12	ON	--	ON	ON	--	ON	50	32	--	--	ON	ON	--	ON
19	13	ON	--	ON	ON	--	--	51	33	--	--	ON	ON	--	--
20	14	ON	--	ON	--	ON	ON	52	34	--	--	ON	--	ON	ON
21	15	ON	--	ON	--	ON	--	53	35	--	--	ON	--	ON	--
22	16	ON	--	ON	--	--	ON	54	36	--	--	ON	--	--	ON
23	17	ON	--	ON	--	--	--	55	37	--	--	ON	--	--	--
24	18	ON	--	--	ON	ON	ON	56	38	--	--	--	ON	ON	ON
25	19	ON	--	--	ON	ON	--	57	39	--	--	--	ON	ON	--
26	1A	ON	--	--	ON	--	ON	58	3A	--	--	--	ON	--	ON
27	1B	ON	--	--	ON	--	--	59	3B	--	--	--	ON	--	--
28	1C	ON	--	--	--	ON	ON	60	3C	--	--	--	--	ON	ON
29	1D	ON	--	--	--	ON	--	61	3D	--	--	--	--	ON	--
30	1E	ON	--	--	--	--	ON	62	3E	--	--	--	--	--	ON
31	1F	ON	--	--	--	--	--	63	3F	--	--	--	--	--	--

第二步把磁碟取出，換上 MOCKINGBOARD 的磁碟（隨咕附有的那一張），按 CTL-C , RETURN 返回 BASIC，然後鍵入 NEW , LOAD PRIM . 再把程式 LIST 出來，將第200行改爲 PRINT CHR $（4）；"BLOAD TABLE ACCESS"，再加上第210行 PRINT CHR $（4）；"RUN TABLE"，取出 MOCKINGBOARD 磁碟再換上已經存放 TABLE ACCESS 的磁碟，把改好的程式以 PRIM 名 SAVE 到磁碟上。

第三步再放回 MOCKINGBOARD 磁碟，鍵入 NEW 清除舊程式，然後 LOAD TABLE，把程式 LIST 出來，鍵入以下加上的部份：

```
140 A = 33024: REM  BEIGINNING AD
     DRES OF TABLE
150  FOR X = 0 TO 15: POKE A + X,
     0: NEXT
160  POKE 8,0: POKE 9,129
170  CALL 32768
180  REM
190  REM
200  HOME : CALL  - 336
```

最後把程式以 TABLE 的名稱 SAVE 到已經有 TABLE ACCESS 和 PRIM 的新磁碟上。

上面幾個程式最主要是把 INIT，LATCH，WRITE 等存入電腦記憶內，因爲原本已經存於 MOCKING-BOARD 磁碟上，所以借用以節省時間。

相信目前最令人難於理解便是 COARSE 和FINE 兩種調節。就讓我們立即觀察這兩種調節的效果吧。假如你的磁碟（新的那一張）還留在磁碟機內，請鍵入 NEW 清除記憶

，然後鍵入 RUN PRIM，待所有程式輸入電腦後，螢幕上畫面會被清除而游標會跳到左上角。現在請鍵入下列程式，所有 REM 都是參考之用，可以省去不鍵入。

```
10   REM  ** PROGRAM TO OBSERVE **

20   REM  ** THE DIFFERENCE OF  **

30   REM  ** COARSE & FINE TUNE **

40   REM
49   REM  INITIALIZED MOCKINGBOARD

50   CALL 36864: CALL 36908
100 R1 = 252: REM  REGISTER'S VAR
    IABLE OF 1ST SPEAKER
110 R2 = 253: REM  REGISTER'S VAR
    IABLE OF 2ND SPEAKER
120 D1 = 254: REM  REGISTER'S DAT
    A OF 1ST SPEAKER
130 D2 = 255: REM  REGISTER'S DAT
    A OF 2ND SPEAKER
140 S1 = 36608: REM  REGISTER'S A
    CCESS ROUTINE OF 1ST SPEAKER

150 S2 = 36625: REM  REGISTER'S A
    CCESS ROUTINE  OF 2ND SPEAKE
    R
160 R3 = 36897: REM  RESET 1ST SP
    EAKER
170 R4 = 36941: REM  RESET 2ND SP
    EAKER
180  POKE R1,7: POKE D1,62: CALL
    S1: POKE R1,8: POKE D1,12: CALL
    S1: POKE R1,13: POKE D1,9: CALL
    S1
```

```
190  HOME :N = 0:M = 0
200  POKE R1,1: POKE D1,M: CALL S
     1: POKE R1,0: POKE D1,N: CALL
     S1
210 N = N + 1: IF N = 255 THEN  HOME
     :M = M + 1:N = 0
220  IF M > 255 THEN 240
230  VTAB 10: HTAB 2: PRINT "CRSE
     = ";M;"  ";"FINE= ";N: GOTO
     200
240  CALL R3
250  END
```

當 RUN 這個示範程式時，畫面上同時顯示出 CRSE 和FINE 的數值，隨着數值的變化，聲音由高的音調一直下降。FINE 的數值由零開始一直加1至255爲止，然後 CRSE 的值加一，這時 FINE 會由零從新開始。換言之，FINE 調節是把 CRSE 每一個數值分爲255個單位去調節。以時鐘舉例，則 CRSE 等於分針，而 FINE 則等於秒針。相信大家都留意到當 CRSE 數值大過零之後，音調一直是低沉而且開始沒有顯著的變化。更有趣的是，當 CRSE 數值加到16時，音調又重新由高音開始。結論是一般的樂音，除特殊需要外，CRSE 都處於零和一的數值，而 FINE 會是較多的調節，倘若要更低音的樂音，則 CRSE 便調節較大了。

這時你或許會問，爲甚麼不直接用 MOCKING-BOARD 的 SOUND UTILITY 測試？問題的答案很明顯，SOUND UTILITY 每次按 P 時電腦會把整個 TABLE 重新輸入一次，這是不必要的步驟。基本上 SOUND UTILITY 是供應我們去編製聲音效果之用，它現有的效果例如機器聲、海浪聲、槍聲等亦是給予我們作爲參考，但假如一些較複雜的效果，例如幾種不同的聲音組合，SOUND

UTILITY 便較難於表達了。

回頭再研究我們的示範程式。TABLE ACCESS 這個檔案，亦是你第一次鍵入的機器碼，其實應該稱之爲 REGISTER ACCESS 比較貼切。因爲它只是把其中之任何一個所需 REGISTER 輸入資料而已。請看看 TABLE 的程式，最後加入的第140至170行是把零的數值存入所有 REGISTER 內，表示我們只把 REGISTER 的工作編排好，但不輸入聲音資料，如果這時你呼叫任何一個 MOCKINGBOARD 的基本程式，都不會產生聲音。直至你現在已經鍵入的示範程式第180行才正式供給資料。POKE R1, 7表示應用第七個記存器，POKE D1, 62表示 ENABLE（參看上面圖2.1）是62，亦即是只開啓第一個喇叭的第一個純樂音聲道，其他聲道和噪音都暫時不需要。POKE R1, 8表示應用第8個記存器，而 POKE D1, 12表示聲量單位是12，如此類推。

到第200行開始是聲音的產生了，M 和 N 是變化的數值，N 由零開始加1直至255時則 M 便會加一，而 N 會由零從新開始，這是 FINE 和 COARSE 的數值。順理成章 POKE R1, 1和 POKE R1, 0便是第一和零的 REGISTER。如此我們便無須每次把 TABLE 輸入一次，而是只須輸入要用的資料便可，這對於幾種不同聲音效果的產生十分重要。至於 S1和 S2是呼叫 TABLE ACCESS 的兩個喇叭位址，目前只應用 S1來觸動第一個喇叭。

請把程式第180行開始全部刪去，鍵入 DEL 180, 250，然後再鍵入下列部份：

```
175  POKE R1,1: POKE D1,0: CALL S
     1: POKE R1,3: POKE D1,0: CALL
     S1
180  POKE R1,7: POKE D1,60: CALL
     S1
185  POKE R1,8: POKE D1,12: CALL
     S1: POKE R1,9: POKE D1,12: CALL
     S1
186  POKE R1,13: POKE D1,9: CALL
     S1
190  FOR I = 0 TO 255
200  POKE R1,0: POKE D1,I: CALL S
     1: POKE R1,2: POKE D1,255 -
     I: CALL S1
210  NEXT I
220  FOR I = 255 TO 0 STEP  - 1
230  POKE R1,0: POKE D1,I: CALL S
     1: POKE R1,2: POKE D1,255 -
     I: CALL S1
240  NEXT I
250  GOTO 190
```

按 RUN 運行程式時，第一個喇叭的第一和第二聲道同時產生樂音了。第一聲道由高至低而第二聲道由低至高不斷發聲。上面第180行我們 POKE R1, 7和 POKE D1, 60開啓第一和第二聲道，當然還要加上 POKE R1, 9和 POKE D1, 12使第二聲道的音量開啓了。

第190行我們把聲音資料的 FINE 調節由零至255逐一輸入第一聲道而相反地由255至零輸入第二聲道，至於第220行則由255至零輸入第一聲道而第二聲道卻變成由零至255，如此程式週環運行便產生兩個聲道的效果了。

現在按 CTL-RESET 把運行停止，讓我們看看可否加

上一些效果。加入第176行並且把第180和186行更改如下：

```
176  POKE R1,4: POKE D1,200: CALL
     S1: POKE R1,12: POKE D1,2: CALL
     S1
180  POKE R1,7: POKE D1,24: CALL
     S1: POKE R1,10: POKE D1,2: CALL
     S1
186  POKE R1,13: POKE D1,8: CALL
     S1
```

請鍵入 RUN ，除了原有的樂音效果外，更加添上鼓聲效果了。因爲第176行我們再開啓第三聲道，而 POKE R1, 12則爲鼓聲加上一個音形的調節。第180行我們 POKE R1, 7和 POKE D1, 24使第三聲道同時產生樂音和噪音，第186行加上音形（ENVELOPE SHAPE），數值是8（參考圖2. 2 及圖2. 3.）

聲音的混合

一些特殊的聲音效果，很多時不能單靠 SOUND UTILITY 便可做到。下面的程式是空山鳥語的效果，請把第175行起所有行號程式刪去，然後加入：

```
175  POKE R1,1: POKE D1,0: CALL S
     1: POKE R1,3: POKE D1,0: CALL
     S1: POKE R1,5: POKE D1,0: CALL
     S1
180  POKE R1,7: POKE D1,56: CALL
     S1
185  POKE R1,8: POKE D1,10: CALL
     S1: POKE R1,9: POKE D1,10: CALL
     S1: POKE R1,10: POKE D1,10: CALL
     S1
186  POKE R1,13: POKE D1,0: CALL
     S1
190 N = 0:M = 0:O = 0
200  POKE R1,0: POKE D1,N: CALL S
     1: POKE R1,2: POKE D1,M: CALL
     S1: POKE R1,4: POKE D1,O: CALL
     S1
210 N = N + 1: IF N > 10 THEN N =
      INT ( RND (1) * 5)
220 M = M + 1: IF M > 25 THEN M =
      INT ( RND (1) * 20)
230 O = O + 1: IF O > 30 THEN O =
      INT ( RND (1) * 25)
240  GOTO 200
```

相信現在不用再解釋R1和D1的用途了，不過鳥兒叫鳴的效果是在高音域部份不斷把音調滑上或滑落所造成。假如讀者曾經學習小提琴或二胡之類的絃樂器，都知道在最高音域上使用不同滑音便會產生鳥鳴效果。這裡請看第190行，N，M和O三個變數代表三個聲道上的滑音數值，全部聲音都是由零的值向下滑行，亦等於由最高音域向下滑行。第185行我們已經總共開了三個聲道。

由於鳥兒的叫鳴是雜亂無章的，所以每次滑奏完畢，我們在第210至230行加入亂數，讓鳥鳴的音域不斷變化，如此一種百鳥投林的效果便產生了。

讓我們繼續把鳥鳴再加入槍聲如何。在 SOUND UTILITY 中，倘若我們按 L（LOAD）GUNSHOT，會有如下顯示：

```
REGISTER                MAX       CHANNEL
NAME                    VAL  ALL   A      B      C
==================================================
TONE PER FINE           255  0     0      0      0
TONE PER COARSE          15  0     0      0      0
NOISE PERIOD             31  16
ENABLE                   63  7
AMPLITUDE                16  0     16     16     16
(FIX=0-15/VAR=16)
ENVL PER FINE           255  0
ENVL PER COARSE         255  10
ENVL SHAPE               15  0
==================================================
```

換言之，槍聲是利用音形數值零來產生，而 ENABLE 數值7表示只應用噪音部份，ENVL PER COARSE 的數值是10，和純音的調節一樣，噪音亦同理有 COARSE 和 FINE 兩種調節，不過效果卻不是指音域而言；因爲噪音並不會有高低音調。這裡的調節是針對 ENVL SHAPE 而設。

我們要明白一點的是音形只能在音量（AMPLITUDE）數值是16時才能發揮效果，其他音量數值都不受音形影响。音量數值16表示聲音產生效果由音形所決定。這些音形可以由圖2.2觀察出來，但音形並不是固定的，意思是聲音雖然隨著音形的高低產生，但它可以在最短時間內把整個音形

全部做完，或者在很長的時間去完結。舉例而言好像第8號的音形，聲音好像彈簧一樣由高至低不斷重複，但這條彈簧可以是一條壓縮的彈簧，亦可以是一條鬆散的彈簧，這就靠COARSE 和 FINE 來調節了。

請試試把不同數值加入 COARSE 調節內，你會發覺數值越大，則聲音的餘响會加長。

在我們一直作示範的程式第175行開始刪去，加入下面程式：

```
175  POKE R1,1: POKE D1,0: CALL S
     1: POKE R1,6: POKE D1,16: CALL
     S1
180  POKE R1,7: POKE D1,14: CALL
     S1
185  POKE R1,8: POKE D1,12: CALL
     S1
190 N = 0
200  POKE R1,0: POKE D1,N: CALL S
     1
210 N = N + 1: IF N > 25 THEN N =
      INT ( RND (1) * 18) + 6
240  IF  PEEK ( - 16287) > 127 THEN
     300
245 F = 0
250  GOTO 200
300  IF F = 1 THEN  GOTO 200
310 F = 1: POKE R1,12: POKE D1,10
     : CALL S1: POKE R1,13: POKE
     D1,0: CALL S1: POKE R1,9: POKE
     D1,16: CALL S1: POKE R1,10: POKE
     D1,16: CALL S1: GOTO 200
```

鍵入 RUN，一隻雀鳥在不斷飛來飛去鳴叫，倘若我們按 JOYSTICK 的 BUTTON 零，則會有槍聲效果，好像射擊鳥兒一樣。

第175至185行開啓所需聲道，我們選擇第一聲道作爲鳥鳴，第二和第三聲道作爲槍聲，所以第6的 REGISTER 我們給予 NOISE PERIOD 一個16的數値。而第7個 REGISTER 我們給予14數値，表示採用第一聲道的純音部份，第二和第三聲道的噪音部份。

第200行的 N 變數是鳥鳴滑音變數，第240行測試是否按鈕，第245行的 F＝0 是測試我們是否按着不放，倘若按鈕被長時期按著不放的話，則 F 會變爲1而不發槍聲。第310行便是把槍聲的所有資料輸入，則槍聲便會產生了。

讀者可能留意到程式由始至終沒有用到 RESET 程式，因爲 RESET 會把所有記存器變爲零，這不符合要求。倘若呼叫 RESET，鳥鳴便會忽然停下來，再斷續產生，造成繼續的感覺，事實上，槍聲產生後會自動消除，這是基於 ENVELOPE SHAPE 的影响。

最後，讓我們把圖形配上鳥鳴和槍聲作爲結束本章。下列程式以 APPLESOFT 圖形表編成，畫面上有一隻叫鳴的雀鳥飛來飛去，畫面底部有一個人形圖案。按 BUTTON 零便會有子彈射出。程式運行當然比較慢，因爲是 BASIC 所編寫，讀者如果要得到更佳效果，可以用 TASC 或其他 COMPILER 把它轉爲機械語，則效果變爲快了數十倍之多，不過要留意記憶體的編排。如果用 TASC 的話，OBJECT PROGRAM 必須放在 $ 5000開始。

```
1   REM  *****************
2   REM  * SINGING BIRD *
3   REM  *****************
10  PRINT  CHR$ (4);"BLOAD ST.MAN
      & BIRD"
15  PRINT  CHR$ (4);"BLOAD BIRD.P
     TH"
20  POKE 232,0: POKE 233,96
25  GOSUB 2000
30  SCALE= 1: ROT= 0: HCOLOR= 3: HGR
     : POKE  - 16302,0
35  XDRAW 4 AT 140,170
40 N% = 1:X% = 20:Y% = 20:AN% = 0
     :PD% = 16384
45 F = 0:XB% = 148:YB% = 150
50  XDRAW N% AT X%,Y%:NO% = N%:XO
     % = X%:YO% = Y%
54  IF YB% < 10 THEN F = 0: XDRAW
     5 AT XB%,YB%:XB% = 140:YB% =
     166: GOTO 57
55  IF F = 1 THEN  XDRAW 5 AT XB%
     ,YB%:YB% = YB% - 4: XDRAW 5 AT
     XB%,YB%: GOTO 70
57  IF  PEEK ( - 16287) < 128 THEN  70
59  POKE R1%,12: POKE D1%,10: CALL
     S1: POKE R1%,13: POKE D1%,0:
      CALL S1: POKE R1%,9: POKE D
     1%,16: CALL S1: POKE R1%,10:
      POKE D1%,16: CALL S1
60 XB% = 152:YB% = 150: XDRAW 5 AT
     XB%,YB%:F = 1
70 N% = N% + 1: IF N% > 3 THEN N%
      = 1
75  POKE R1%,0: POKE D1%,BN%: CALL
     S1:BN% = BN% + 2: IF BN% > 2
     0 THEN BN% =  INT ( RND (1) *
```

```
     15)
80 AD% = PD% + AN%: GOSUB 1000:X%
      = X% + SX%:Y% = Y% + SY%: XDRAW
     NO% AT XO%,YO%
90  GOTO 50
1000 P% =  PEEK (AD%)
1010  IF P% = 2 THEN SX% = 2:SY% =
     0: GOTO 1090
1020  IF P% = 34 THEN SX% = 2:SY%
      = 2: GOTO 1090
1030  IF P% = 130 THEN SX% = 2:SY
     % =  - 2: GOTO 1090
1040  IF P% = 32 THEN SX% = 0:SY%
      = 2: GOTO 1090
1050  IF P% = 40 THEN SX% =  - 2:
     SY% = 2: GOTO 1090
1060  IF P% = 8 THEN SX% =  - 2:S
     Y% = 0: GOTO 1090
1070  IF P% = 136 THEN SX% =  - 2
     :SY% =  - 2: GOTO 1090
1080  IF P% = 128 THEN SX% = 0:SY
     % =  - 2: GOTO 1090
1085 AN% = 0:AD% = 16384: GOTO 10
     00
1090 AN% = AN% + 1: RETURN
2000  CALL 36864: CALL 36908
2010 R1% = 252:R2% = 253:D1% = 25
     4:D2% = 255:S1 = 36608:S2 =
     36625
2020  POKE R1%,1: POKE D1%,0: CALL
     S1: POKE R1%,6: POKE D1%,16:
      CALL S1: POKE R1%,7: POKE D
     1%,14: CALL S1: POKE R1%,8: POKE
     D1%,15: CALL S1
2030 BN% = 0
2040  RETURN
```

第三章

MUSIC CONSTRUCTION SET 的特殊應用

相信大家都知道有一個軟件名爲 MUSIC CONSTR-UC TION SET，當它配合 MOCKINGBARD 可以編成一首美妙的音樂以供欣賞。

一直以來，電腦都被視爲一件冰冷的電子機器，就算電子琴之類的樂器，都被一般學習正統音樂的人認爲是電子音樂而已，不容易作爲富有感情的樂器，但音樂編製器這個軟件竟然可以產生豐富的音色，充份發揮音樂的優點，使藝術和電腦之間得以溝通。

MUSIC CONSTRUCTION SET 除了給我們作爲欣賞音樂之外，可以說是妙用無窮。筆者曾經在電腦時代第16和17期討論過如何利用這個軟件把任何程式配上背景音樂，這裡不打算再重複了。現在所談的就是，旣然程式可以配上背景音樂，如果我們希望把背景音樂、聲音效果，一同配上程式上又怎樣呢？

要編造這樣複雜的工作，讀者必須對 MUSIC CON-STRUCTION SET 有較深入的了解，有興趣的朋友倘若未對該軟件有所認識的話，請先參閱第16和17期的關於 MUSIC CONSTRUCTION SET 背景音樂，才能對以下所討論項目明白。

首先，讓我重溫一下該軟件關於音樂產生部份的 SOURCE CODE，參看 LISTING ＃3.1

＊如發覺聲音有雜聲，請盡量採用這個檔案在自己程式內。＊

```
*
                  1                                ;"WILL HARVEY'S MUSIC"
                  2                                ;"COPYRIGHT (C) 1983"
                  3                    ORG   $4900
                  4   ORB              EQU   $C400
                  5   ORA              EQU   $C401
                  6   ORB2             EQU   $C480
                  7   ORA2             EQU   $C481
                  8   DDRB2            EQU   $C482
                  9   DDRA2            EQU   $C483
                  10  DDRB             EQU   $C402
                  11  DDRA             EQU   $C403
                  12  SONGADD          EQU   $4
                  13  CHANNADD         EQU   $D6
4900: 4C B1 49    14                   JMP   INTRUPT
4903: 4C 7C 4B    15                   JMP   INIT
4906: 4C C8 4A    16                   JMP   INITMOCK
4909: 4C 3D 4B    17                   JMP   SONGADDS
490C: 4C D8 4B    18                   JMP   PAUSE
490F: 4C EB 4B    19                   JMP   CONTINUE
4912: 03          20  TEMPO            HEX   03
4913: 03          21  DECAY            HEX   03
4914: FE 3F 7E
4917: 44          22  STARTADD         HEX   FE3F7E44
4918: 0A          23  STRONG           HEX   0A
4919: 00          24  START            HEX   00
491A: 06          25  END              HEX   06
491B: 0A          26  REST             HEX   0A
491C: 00          27  TEMPCNTR         HEX   00
491D: 00          28  DECCNTR          HEX   00
491E: 00          29  TEMP5            HEX   00
491F: 00          30  TEMP0            HEX   00
4920: 00          31  TEMP1            HEX   00
4921: 00          32  TEMP2            HEX   00
4922: 00          33  TEMP4            HEX   00
4923: 00          34  TEMP6            HEX   00
4924: 01 01       35  CNTR             HEX   0101
4926: FF FF FF
4929: FF FF FF    36  VOICE            HEX   FFFFFFFFFFFF
492C: 00          37  PLUG             HEX   00
492D: 00          38  FREQLO           HEX   00
492E: 00          39  FREQHI           HEX   00
492F: 00 00       40  TIED             HEX   0000
4931: 1E 1F 20
4934: 22 24 26
4937: 29 2C       41  FREQLOS          HEX   1E1F20222426292C
4939: 2E 30 33
493C: 36 3A 3D
493F: 41 45       42                   HEX   2E3033363A3D4145
4941: 49 4D 52
4944: 56 5C 61
4947: 67 6D       43                   HEX   494D52565C61676D
4949: 73 7A 81
494C: 89 91 9A
494F: A3 AD       44                   HEX   737A8189919AA3AD
4951: B7 C2 CE
4954: DA E7 F4
4957: 03 12       45                   HEX   B7C2CEDAE7F40312
4959: 23 34 46
495C: 5A 6E 84
```

```
495F: 9B B3      46          HEX  2334465A6E849BB3
4961: CD E9 06
4964: 25 45 68
4967: 8C B3      47          HEX  CDE9062545688CB3
4969: DC 08 36
496C: 67 9B D2
496F: 01 01      48          HEX  DC0836679BD20101
4971: 00 00 00
4974: 00 00 00
4977: 00 00      49  FREQHIS HEX  0000000000000000
4979: 00 00 00
497C: 00 00 00
497F: 00 00      50          HEX  0000000000000000
4981: 00 00 00
4984: 00 00 00
4987: 00 00      51          HEX  0000000000000000
4989: 00 00 00
498C: 00 00 00
498F: 00 00      52          HEX  0000000000000000
4991: 00 00 00
4994: 00 00 00
4997: 01 01      53          HEX  0000000000000101
4999: 01 01 01
499C: 01 01 01
499F: 01 01      54          HEX  0101010101010101
49A1: 01 01 02
49A4: 02 02 02
49A7: 02 02      55          HEX  0101020202020202
49A9: 02 03 03
49AC: 03 03 03
49AF: 00 00      56          HEX  0203030303030000
49B1: 8A         57  INTRUPT TXA
49B2: 48         58          PHA
49B3: 98         59          TYA
49B4: 48         60          PHA
49B5: A9 C0      61          LDA  #%11000000
49B7: 8D 0D C4   62          STA  $C40D
49BA: EE 1D 49   63  CHANCE  INC  DECCNTR
49BD: EE 1C 49   64          INC  TEMPCNTR
49C0: AD 1D 49   65          LDA  DECCNTR
49C3: CD 13 49   66          CMP  DECAY
49C6: D0 08      67          BNE  NODECAY
49C8: 20 A0 4B   68          JSR  DIMINISH
49CB: A9 00      69          LDA  #0
49CD: 8D 1D 49   70          STA  DECCNTR
49D0: AD 1C 49   71  NODECAY LDA  TEMPCNTR
49D3: CD 12 49   72          CMP  TEMPO
49D6: F0 03      73          BEQ  MUSIC
49D8: 4C 6F 4A   74          JMP  MUSICRTI
49DB: A9 00      75  MUSIC   LDA  #$0
49DD: 8D 1C 49   76          STA  TEMPCNTR
49E0: A2 00      77          LDX  #0
49E2: 8E 21 49   78  MUSIC12 STX  TEMP2
49E5: 8A         79          TXA
49E6: 0A         80          ASL
49E7: 8D 22 49   81          STA  TEMP4
49EA: DE 24 49   82          DEC  CNTR,X
```

```
49ED: BD 24 49   83            LDA  CNTR,X
49F0: D0 6F      84            BNE  MUSIC9
49F2: 20 1E 4B   85            JSR  STOPVOIC
49F5: AE 22 49   86   MUSIC15  LDX  TEMP4
49F8: B5 04      87            LDA  SONGADD,X
49FA: 18         88            CLC
49FB: 69 02      89            ADC  #2
49FD: 95 04      90            STA  SONGADD,X
49FF: 90 02      91            BCC  MUSIC10
4A01: F6 05      92            INC  SONGADD+1,X
4A03: A1 04      93   MUSIC10  LDA  (SONGADD,X)
4A05: 4A         94            LSR
4A06: 8D 1F 49   95            STA  TEMP0
4A09: F6 04      96            INC  SONGADD,X
4A0B: A1 04      97            LDA  (SONGADD,X)
4A0D: D6 04      98            DEC  SONGADD,X
4A0F: 8D 20 49   99            STA  TEMP1
4A12: 0D 1F 49   100           ORA  TEMP0
4A15: D0 06      101           BNE  MUSIC11
4A17: 20 3D 4B   102           JSR  SONGADDS
4A1A: 4C 6C 4A   103           JMP  MUSIC3
4A1D: AE 21 49   104  MUSIC11  LDX  TEMP2
4A20: 20 76 4A   105           JSR  SEARCH
4A23: AD 21 49   106           LDA  TEMP2
4A26: 9D 26 49   107           STA  VOICE,X
4A29: 8E 23 49   108           STX  TEMP6
4A2C: AE 1F 49   109           LDX  TEMP0
4A2F: BD 31 49   110           LDA  FREQLOS,X
4A32: 8D 2D 49   111           STA  FREQLO
4A35: BD 71 49   112           LDA  FREQHIS,X
4A38: 8D 2E 49   113           STA  FREQHI
4A3B: AD 18 49   114           LDA  STRONG
4A3E: 8D 2C 49   115           STA  PLUG
4A41: AE 23 49   116           LDX  TEMP6
4A44: 20 9A 4A   117           JSR  PLUGIT
4A47: AE 21 49   118  MUSIC7   LDX  TEMP2
4A4A: AD 20 49   119           LDA  TEMP1
4A4D: 29 40      120           AND  #$40
4A4F: 9D 2F 49   121           STA  TIED,X
4A52: AD 20 49   122           LDA  TEMP1
4A55: 29 3F      123           AND  #$3F
4A57: 9D 24 49   124           STA  CNTR,X
4A5A: AD 20 49   125           LDA  TEMP1
4A5D: 29 80      126           AND  #$80
4A5F: D0 94      127           BNE  MUSIC15
4A61: AE 21 49   128  MUSIC9   LDX  TEMP2
4A64: E8         129           INX
4A65: E0 02      130           CPX  #$2
4A67: F0 03      131           BEQ  MUSIC3
4A69: 4C E2 49   132           JMP  MUSIC12
4A6C: 20 C8 4A   133  MUSIC3   JSR  INITMOCK
4A6F: 68         134  MUSICRTI PLA
4A70: A8         135           TAY
4A71: 68         136           PLA
4A72: AA         137           TAX
4A73: A5 45      138           LDA  $45
4A75: 40         139           RTI
```

```
4A76: E0 00      140  SEARCH   CPX  #$0
4A78: D0 10      141           BNE  S2
4A7A: AE 19 49   142           LDX  START
4A7D: BD 26 49   143  S4       LDA  VOICE,X
4A80: 30 17      144           BMI  S3
4A82: E8         145           INX
4A83: EC 1A 49   146           CPX  END
4A86: D0 F5      147           BNE  S4
4A88: CA         148           DEX
4A89: 60         149           RTS
4A8A: AE 1A 49   150  S2       LDX  END
4A8D: CA         151           DEX
4A8E: BD 26 49   152  S5       LDA  VOICE,X
4A91: 30 06      153           BMI  S3
4A93: CA         154           DEX
4A94: EC 19 49   155           CPX  START
4A97: D0 F5      156           BNE  S5
4A99: 60         157  S3       RTS
4A9A: 8A         158  PLUGIT   TXA
4A9B: 48         159           PHA
4A9C: 98         160           TYA
4A9D: 48         161           PHA
4A9E: A9 00      162           LDA  #0
4AA0: E0 03      163           CPX  #3
4AA2: 90 05      164           BCC  PI2
4AA4: CA         165           DEX
4AA5: CA         166           DEX
4AA6: CA         167           DEX
4AA7: A9 10      168           LDA  #$10
4AA9: 85 D6      169  PI2      STA  CHANNADD
4AAB: 8A         170           TXA
4AAC: 0A         171           ASL
4AAD: A8         172           TAY
4AAE: AD 2D 49   173           LDA  FREQLO
4AB1: 91 D6      174           STA  (CHANNADD),Y
4AB3: C8         175           INY
4AB4: AD 2E 49   176           LDA  FREQHI
4AB7: 91 D6      177           STA  (CHANNADD),Y
4AB9: 8A         178           TXA
4ABA: 18         179           CLC
4ABB: 69 08      180           ADC  #$8
4ABD: A8         181           TAY
4ABE: AD 2C 49   182           LDA  PLUG
4AC1: 91 D6      183           STA  (CHANNADD),Y
4AC3: 68         184           PLA
4AC4: A8         185           TAY
4AC5: 68         186           PLA
4AC6: AA         187           TAX
4AC7: 60         188           RTS
4AC8: 98         189  INITMOCK TYA
4AC9: 48         190           PHA
4ACA: A9 FF      191           LDA  #$FF
4ACC: 8D 03 C4   192           STA  DDRA
4ACF: 8D 83 C4   193           STA  DDRA2
4AD2: A9 07      194           LDA  #$7
4AD4: 8D 02 C4   195           STA  DDRB
4AD7: 8D 82 C4   196           STA  DDRB2
```

```
4ADA: A0 00     197              LDY   #$0
4ADC: 8C 01 C4  198   IM2        STY   ORA
4ADF: A9 07     199              LDA   #$7
4AE1: 8D 00 C4  200              STA   ORB
4AE4: A9 04     201              LDA   #$4
4AE6: 8D 00 C4  202              STA   ORB
4AE9: B9 00 03  203              LDA   $300,Y
4AEC: 8D 01 C4  204              STA   ORA
4AEF: A9 06     205              LDA   #$6
4AF1: 8D 00 C4  206              STA   ORB
4AF4: A9 04     207              LDA   #$4
4AF6: 8D 00 C4  208              STA   ORB
4AF9: 8C 81 C4  209              STY   ORA2
4AFC: A9 07     210              LDA   #$7
4AFE: 8D 80 C4  211              STA   ORB2
4B01: A9 04     212              LDA   #$4
4B03: 8D 80 C4  213              STA   ORB2
4B06: B9 10 03  214              LDA   $310,Y
4B09: 8D 81 C4  215              STA   ORA2
4B0C: A9 06     216              LDA   #$6
4B0E: 8D 80 C4  217              STA   ORB2
4B11: A9 04     218              LDA   #$4
4B13: 8D 80 C4  219              STA   ORB2
4B16: C8        220              INY
4B17: C0 0F     221              CPY   #$F
4B19: D0 C1     222              BNE   IM2
4B1B: 68        223              PLA
4B1C: A8        224              TAY
4B1D: 60        225              RTS
4B1E: AE 19 49  226   STOPVOIC   LDX   START
4B21: BD 26 49  227   SV2        LDA   VOICE,X
4B24: CD 21 49  228              CMP   TEMP2
4B27: D0 0D     229              BNE   SV3
4B29: A9 FF     230              LDA   #$FF
4B2B: 9D 26 49  231              STA   VOICE,X
4B2E: A9 00     232              LDA   #$0
4B30: 8D 2C 49  233              STA   PLUG
4B33: 20 9A 4A  234              JSR   PLUGIT
4B36: E8        235   SV3        INX
4B37: EC 1A 49  236              CPX   END
4B3A: D0 E5     237              BNE   SV2
4B3C: 60        238              RTS
4B3D: AD 14 49  239   SONGADDS   LDA   STARTADD
4B40: 85 04     240              STA   SONGADD
4B42: AD 15 49  241              LDA   STARTADD+1
4B45: 85 05     242              STA   SONGADD+1
4B47: AD 16 49  243              LDA   STARTADD+2
4B4A: 85 06     244              STA   SONGADD+2
4B4C: AD 17 49  245              LDA   STARTADD+3
4B4F: 85 07     246              STA   SONGADD+3
4B51: A9 00     247              LDA   #$0
4B53: 8D FE 03  248              STA   $3FE
4B56: A9 49     249              LDA   #$49
4B58: 8D FF 03  250              STA   $3FF
4B5B: A9 00     251              LDA   #$0
4B5D: 85 D6     252              STA   CHANNADD
4B5F: A9 03     253              LDA   #$3
```

```
4B61: 85 D7       254           STA  CHANNADD+1
4B63: A9 01       255           LDA  #$1
4B65: 8D 24 49    256           STA  CNTR
4B68: 8D 25 49    257           STA  CNTR+1
4B6B: A9 00       258           LDA  #$0
4B6D: 8D 21 49    259           STA  TEMP2
4B70: 20 1E 4B    260           JSR  STOPVOIC
4B73: A9 01       261           LDA  #$1
4B75: 8D 21 49    262           STA  TEMP2
4B78: 20 1E 4B    263           JSR  STOPVOIC
4B7B: 60          264           RTS
4B7C: 20 3D 4B    265 INIT      JSR  SONGADDS
4B7F: A9 F8       266           LDA  #$F8
4B81: 8D 07 03    267           STA  $307
4B84: 8D 17 03    268           STA  $317
4B87: A9 40       269           LDA  #%01000000
4B89: 8D 0B C4    270           STA  $C40B
4B8C: A9 C0       271           LDA  #%11000000
4B8E: 8D 0D C4    272           STA  $C40D
4B91: 8D 0E C4    273           STA  $C40E
4B94: A9 FF       274           LDA  #$FF
4B96: 8D 04 C4    275           STA  $C404
4B99: A9 40       276           LDA  #$40
4B9B: 8D 05 C4    277           STA  $C405
4B9E: 58          278           CLI
4B9F: 60          279           RTS
4BA0: A2 00       280 DIMINISH  LDX  #$00
4BA2: BD 26 49    281 D1        LDA  VOICE,X
4BA5: C9 02       282           CMP  #$2
4BA7: B0 11       283           BCS  D2
4BA9: A8          284           TAY
4BAA: B9 2F 49    285           LDA  TIED,Y
4BAD: D0 0B       286           BNE  D2
4BAF: BD 08 03    287           LDA  $308,X
4BB2: CD 1B 49    288           CMP  REST
4BB5: F0 03       289           BEQ  D2
4BB7: DE 08 03    290           DEC  $308,X
4BBA: BD 29 49    291 D2        LDA  VOICE+3,X
4BBD: C9 02       292           CMP  #$2
4BBF: B0 11       293           BCS  D3
4BC1: A8          294           TAY
4BC2: B9 2F 49    295           LDA  TIED,Y
4BC5: D0 0B       296           BNE  D3
4BC7: BD 18 03    297           LDA  $318,X
4BCA: CD 1B 49    298           CMP  REST
4BCD: F0 03       299           BEQ  D3
4BCF: DE 18 03    300           DEC  $318,X
4BD2: E8          301 D3        INX
4BD3: E0 03       302           CPX  #$3
4BD5: D0 CB       303           BNE  D1
4BD7: 60          304           RTS
4BD8: 78          305 PAUSE     SEI
4BD9: A9 00       306           LDA  #$00
4BDB: 8D 21 49    307           STA  TEMP2
4BDE: 20 1E 4B    308           JSR  STOPVOIC
4BE1: EE 21 49    309           INC  TEMP2
4BE4: 20 1E 4B    310           JSR  STOPVOIC
```

```
4BE7: 20 C8 4A   311             JSR   INITMOCK
4BEA: 60         312             RTS
4BEB: 58         313   CONTINUE CLI
4BEC: 60         314             RTS

--End assembly--

749 bytes

Errors: 0

Symbol table - alphabetical order:

?  CHANCE   =$49BA     CHANNADD=$D6       CNTR     =$4924     CONTINUE=$4BEB
   D1       =$4BA2     D2      =$4BBA     D3       =$4BD2     DDRA    =$C403
   DDRA2    =$C483     DDRB    =$C402     DDRB2    =$C482     DECAY   =$4913
   DECCNTR  =$491D     DIMINISH=$4BA0     END      =$491A     FREQHI  =$492E
   FREQHIS  =$4971     FREQLO  =$492D     FREQLOS  =$4931     IM2     =$4ADC
   INIT     =$4B7C     INITMOCK=$4AC8     INTRUPT  =$49B1     MUSIC   =$49DB
   MUSIC10  =$4A03     MUSIC11 =$4A1D     MUSIC12  =$49E2     MUSIC15 =$49F5
   MUSIC3   =$4A6C  ?  MUSIC7  =$4A47     MUSIC9   =$4A61     MUSICRTI=$4A6F
   NODECAY  =$49D0     ORA     =$C401     ORA2     =$C481     ORB     =$C400
   ORB2     =$C480     PAUSE   =$4BD8     PI2      =$4AA9     PLUG    =$492C
   PLUGIT   =$4A9A     REST    =$491B     S2       =$4A8A     S3      =$4A99
   S4       =$4A7D     S5      =$4A8E     SEARCH   =$4A76     SONGADD =$04
   SONGADDS=$4B3D      START   =$4919     STARTADD=$4914      STOPVOIC=$4B1E
   STRONG   =$4918     SV2     =$4B21     SV3      =$4B36     TEMPO   =$491F
   TEMP1    =$4920     TEMP2   =$4921     TEMP4    =$4922  ?  TEMP5   =$491E
   TEMP6    =$4923     TEMPCNTR=$491C     TEMPO    =$4912     TIED    =$492F
   VOICE    =$4926

Symbol table - numerical order:

   SONGADD  =$04       CHANNADD=$D6       TEMPO    =$4912     DECAY   =$4913
   STARTADD=$4914      STRONG  =$4918     START    =$4919     END     =$491A
   REST     =$491B     TEMPCNTR=$491C     DECCNTR  =$491D  ?  TEMP5   =$491E
   TEMPO    =$491F     TEMP1   =$4920     TEMP2    =$4921     TEMP4   =$4922
   TEMP6    =$4923     CNTR    =$4924     VOICE    =$4926     PLUG    =$492C
   FREQLO   =$492D     FREQHI  =$492E     TIED     =$492F     FREQLOS =$4931
   FREQHIS  =$4971     INTRUPT =$49B1  ?  CHANCE   =$49BA     NODECAY =$49D0
   MUSIC    =$49DB     MUSIC12 =$49E2     MUSIC15  =$49F5     MUSIC10 =$4A03
   MUSIC11  =$4A1D  ?  MUSIC7  =$4A47     MUSIC9   =$4A61     MUSIC3  =$4A6C
   MUSICRTI=$4A6F      SEARCH  =$4A76     S4       =$4A7D     S2      =$4A8A
   S5       =$4A8E     S3      =$4A99     PLUGIT   =$4A9A     PI2     =$4AA9
   INITMOCK=$4AC8      IM2     =$4ADC     STOPVOIC=$4B1E      SV2     =$4B21
   SV3      =$4B36     SONGADDS=$4B3D     INIT     =$4B7C     DIMINISH=$4BA0
   D1       =$4BA2     D2      =$4BBA     D3       =$4BD2     PAUSE   =$4BD8
   CONTINUE=$4BEB      ORB     =$C400     ORA      =$C401     DDRB    =$C402
   DDRA     =$C403     ORB2    =$C480     ORA2     =$C481     DDRB2   =$C482
   DDRA2    =$C483
```

直接影响音樂的變數

1. 第20行的 TEMPO
負責整首樂曲的速度，數值越大則樂曲演奏越慢
2. 第21行的 DECAY
負責每一個樂音的消逝，一些樂器如鋼琴或結他之類，樂音一經彈奏後，會慢慢消逝。而小提琴或電子琴之類的樂器，則會延長下去。這個數值越大則表示消逝時間較長。
3. 第23行的 STRONG
負責演奏音量。調節音量可以使加入其他聲音效果時避免背景音樂音量過大。
4. 第26行的 REST
和 DECAY 不同，負責樂音的斷續，通常該值不能大過 STRONG。

鳥鳴，槍聲和背景音樂的混合

在 MUSIC CONSTRUCTION SET 內已經存有一個名爲MUSIC 的二進制檔案，以供我們直接取用。該檔案當BLOAD 時會存入＄4000開始的記憶位址上。如果我們發覺該位址有其他用途，則必須修改程式把 MUSIC 移去其他位置。第16期的電腦時代有詳細說明。

我們暫時仍然把 MUSIC 放在＄4000內以供參考之用。

在未實際了解 MUSIC 之前，首先我們預備一首樂曲作爲學習之用。把 MUSIC CONSTRUCTION SET 放入磁碟機內 BOOT 機，如果擁有兩個磁碟機的話，請把一直用作示範的磁碟放進另外一個磁碟機內，如果只有一個磁碟機，則暫時不須理會。讀者可以隨便找一些初級的鋼琴樂譜

作爲編曲，又或者如果有音樂天才，亦可自行作曲。我們現在要編一首高音部和低音部都只是單音的樂曲，亦即高音和低音都分別沒有和絃存在。道理很簡單，每一個樂音交由每邊喇叭其中一個聲道產生，倘若有和絃的話，則會應用其他聲道，但我們希望除了音樂之外，還要加入其他聲音效果，必須預留其他聲道作爲聲音效果之用。

將每一個音符塡在畫面上的五綫譜後，然後把樂曲聆聽一次，如果沒有錯誤的話，則把該首樂曲資料 BSAVE 到另外一個磁碟機上的磁碟內（不要忘記 BSAVE 後加上 D2）。倘若只有一個磁碟機，則需把 MUSIC CONSTRUCTION SET 取出，換上示範磁碟，再作 BSAVE 程序。

跟着請用 FID PROGRAM 把 MUSIC CONSTRUCTION SET 內的 MUSIC 檔案抄錄到示範磁碟內，這時我們可以用示範磁碟作爲研究之用了。如果示範磁碟已經有第二章的所有程式的話，則可以繼續，否則請把上一章提及的程式全部抄錄過來，因爲我們仍然要利用它們產生聲音效果。

按 RUN PRIM，待畫面清除後，請鍵入下列程式：

```
10  REM  ** PROGRAM TO MIX MUSIC
     & SOUND EFFECTS **
35  HOME : VTAB 10: INPUT "NAME O
     F PIECE?";A$
40  HOME : PRINT  CHR$ (4);"BLOAD
      MUSIC": PRINT  CHR$ (4);"BL
     OAD ";A$;",A$4000"
100  POKE 18706,3: POKE 18707,30:
      REM  TEMPO & DECAY
110  POKE 18712,9: REM  STRONG
```

```
120  POKE 18713,2: POKE 18714,4: REM
      USE CHANNEL C OF 1ST SPEAKE
     R & CHANNEL A OF 2ND SPEAKER

130  POKE 18715,9: REM  REST
140  POKE 19328,42: REM  NOISE &
     TONE ENABLE
150 R3 = 36897
160 R4 = 36941
180  FOR I = 768 TO 773: POKE I,0
     : NEXT : REM  CLEAR TABLE DA
     TAS
182  POKE 774,16: POKE 790,16: REM
      NOISE PERIOD VALUE
185  POKE 776,12: POKE 794,12: REM
      1ST SPEAKER CHANNEL A AMPLU
     TUDE & 2ND SPEAKER CHANNEL C
      AMPLUTUDE
188  CALL 18691: REM  MUSIC START
      PLAYING
190 N = 0:F = 0: REM  N=BIRD'S PI
     TCH,F=GUNSHOT DURATION
200  POKE 768,N: POKE 788,N: REM
      INPUT PITCH TO CHANNELS OF
     BOTH SPEAKERS
209  REM  MAKE BIRD'S PITCH RANDO
     M
210 N = N + 1: IF N > 20 THEN N =
      INT ( RND (1) * 10) + 8
220  IF F > 0 THEN 300: REM  HOLD
      GUNSHOT TIME
230  POKE 770,0: POKE 793,0: REM
      STOP GUNSHOT
240  IF  PEEK ( - 16287) > 128 THEN
     F = 17: GOTO 300
```

```
250  REM
260  GOTO 200
300 F = F - 1
310  POKE 777,F: POKE 793,F: REM
      INPUT AMPLITUDE OF GUNSHOT
     TO BOTH SPEAKERS
320  GOTO 200
```

按 RUN 後畫面上會詢問樂曲名稱，輸入後樂曲便開始演奏了。除了樂曲本身之外，我們還會聽到鳥兒叫鳴的聲音，倘若按 BUTTON 零，則跟着會產生槍聲。小心聆聽，我們會發覺所有聲音效果都不會互相影响，亦不會影响樂曲的演奏，這表示 MUSIC CONSTRUCTION SET 除了使我們編製背景音樂之外，更可以產生其他聲音，即使短短幾行的 BASIC 程式都可以做到。最主要的是我們可以盡量把它應用在自己的程式內，發揮它的功用。

現在且看看上面程式的內容，再分析 MUSIC 這個程式。

上面第100和110行是把 TEMPO 和 DECAY 的值輸入 MUSIC 內，讀者可以根據樂曲的速度和音色需要分別輸入，這裡的3和30只供參考之用，因爲筆者是用 IN MAY 這首樂曲作爲示範。如果你編製的樂曲較慢，則請修改 TEMPO 的值以達到效果。

第120行需要較詳細的了解。請參閱 LISTING #3.1 的24和25行，分別爲 START 和 END。這兩個位址內的值十分重要，它們是負責音樂由那個聲道發聲。我們知道樂曲由第一個喇叭至第二個喇叭分別有6個聲道可以同時產生樂音，如果把 START 值加1變爲1而 END 值減1變爲5，則音樂只交由第一喇叭的第二和第三聲道負責，而第二喇叭則由

第一和第二聲道負責，如此第一喇叭和第二喇叭的第一和第三聲道均不會發聲，亦即音樂總共只能產生四個音的和絃，這樣不發聲的聲道便可以給我們作為其他效果之用。如果把 START 和 END 分別增加和減少，最後所有聲道都不能發聲了。

START 和 END 值影響聲道如下表所示：

START VALUE	SPEAKER 1		
	CHANNEL 1	CHANNEL 2	CHANNEL 3
0	ON	ON	ON
1	OFF	ON	ON
2	OFF	OFF	OFF
3	OFF	OFF	OFF

END VALUE	SPEAKER 2		
	CHANNEL 1	CHANNEL 2	CHANNEL 3
6	ON	ON	ON
5	OFF	ON	ON
4	OFF	OFF	OFF
3	OFF	OFF	OFF

表內所影响聲道只是樂音部份，噪音部由於是音樂的關係，一直都被關掉，所以不會有噪音產生，要開啓噪音交由另外一個位址負責。所以第120行我們輸入 START 值2和 END 值4，表示只應用第一喇叭的第三聲道和第二喇叭的

第一聲道作爲樂曲之用，其他聲道留給聲响效果之用。這就是爲甚麼我們編製樂曲時，高音譜和低音譜都只用單音的原因了。

第130行 REST 值設定爲9讓樂曲有連結的演奏。

第140行的19328值亦是重要一環。參看 MUSIC 的第15行，這裡有一條 INIT 的程式，要令樂曲演奏，我們首先便要呼叫18691，亦即這裡開始。讓我們看看第265行的 INIT 是些甚麼東西。4B80內有一個值＄F8，程式會把它放入＄307和＄317的位址內，所有噪音和影响聲音效果的記存器例如 ENVELOPE之類會被關掉，我們把它改爲2A（十進制42）把噪音部份重新開啓，42是開始第二聲道作爲噪音的值（參看上一章）。

筆者曾在16期電腦時代提過＄300開始的第三頁是 MCS 用作和 MOCKINGBOARD 溝通之用，所以第三頁不能任意存放資料。究竟第三頁有些甚麼用途呢？

參看 MUSIC 的第189行 INITMOCK，該處和上一章的 LATCH 和 WRITE 有些相似，事實上，MCS 利用第三頁作爲 TABLE ACCESS。16個記存器的資料首先分別存入＄300和＄310開始位址，第一喇叭使用＄300至＄30F，而第二喇叭使用＄310至＄31F。換言之，改變該處位址資料便可產生聲音效果，而且程式亦無須再作出 INIT 工作了。

所以我們在示範程式的180行首先把＄300至＄31F 輸入零，暫時把兩邊喇叭的所有記存器（REGISTER）清除（如何應用記存器資料請參閱第二章）。

第182行把噪音值存入，如果沒有噪音值，槍聲便不能產生了。

第185行分別存入音量的數值，使鳥鳴的聲音比背景音樂稍大。

第188行開始樂曲的演奏。

第190行至210行相信大家都知道是雀鳥叫聲了。要留意的是 F 這個變數，MUSIC CONSTRUCTION SET 本身要利用 REGISTER 11至13作爲中斷處理，所以原有 MOCKINGBOARD 的 ENVELOP SHAPE 再不能產生效果，因此槍聲要由程式編製。方法是首先在240行測試是否按 BUTTON 零？若是則 F 值爲17，跟着至300行把 F 值作爲音量值減一輸入兩邊喇叭的第二聲道，由於這個聲道只產生噪音，不斷把 F 減一便會產生一種爆炸聲响，直至聲响完畢，第230行便把聲道關掉，如此槍聲可以產生了。

有一點我們必須明白的就是用作示範的程式十分短，所以聲音效果可說絕無問題，但假如程式開始加長的話，則效果便會不符理想了。因爲 BASIC 顯然因速度問題減低效果。試鍵入下列程式把第二章有圖形的程式修改如下：

```
 1  REM  ************************
 2  REM  *   SINGING BIRD        *
 3  REM  ************************
10  PRINT  CHR$ (4);"BLOAD ST.MAN
     & BIRD"
15  PRINT  CHR$ (4);"BLOAD BIRD.P
    TH,A$8000"
16  PRINT  CHR$ (4);"BLOAD MUSIC"
    : PRINT  CHR$ (4);"BLOAD IN
    MAY,A$4000"
20  POKE 232,0: POKE 233,96
25  GOSUB 2000
```

```
30  SCALE= 1: ROT= 0: HCOLOR= 3: HGR
     : POKE  - 16302,0
35  XDRAW 4 AT 140,170
40 N% = 1:X% = 20:Y% = 20:AN% = 0
     :PD = 32768
45 F = 0:XB% = 148:YB% = 150
50  XDRAW N% AT X%,Y%:NO% = N%:XO
     % = X%:YO% = Y%
54  IF YB% < 10 THEN F = 0: XDRAW
     5 AT XB%,YB%:XB% = 140:YB% =
     166: GOTO 57
55  IF F = 1 AND FF > 0 THEN FF =
     FF - 1: POKE 777,FF: POKE 79
     3,FF: XDRAW 5 AT XB%,YB%:YB%
      = YB% - 4: XDRAW 5 AT XB%,Y
     B%: GOTO 70
56  IF F = 1 THEN  POKE 770,0: POKE
     793,0: XDRAW 5 AT XB%,YB%:YB
     % = YB% - 4: XDRAW 5 AT XB%,
     YB%: GOTO 70
57  IF  PEEK ( - 16287) < 128 THEN
     70
59 FF = 14: POKE 77,FF: POKE 793,
     FF
60 XB% = 152:YB% = 150: XDRAW 5 AT
     XB%,YB%:F = 1
70 N% = N% + 1: IF N% > 3 THEN N%
      = 1
75  POKE 768,BN%: POKE 788,BN%:BN
     % = BN% + 1: IF BN% > 20 THEN
     BN% =  INT ( RND (1) * 10) +
     8
80 AD = PD + AN%: GOSUB 1000:X% =
     X% + SX%:Y% = Y% + SY%: XDRAW
     NO% AT XO%,YO%
90  GOTO 50
```

```
1000 P% =  PEEK (AD)
1010  IF P% = 2 THEN SX% = 2:SY% =
     0: GOTO 1090
1020  IF P% = 34 THEN SX% = 2:SY%
      = 2: GOTO 1090
1030  IF P% = 130 THEN SX% = 2:SY
     % =  - 2: GOTO 1090
1040  IF P% = 32 THEN SX% = 0:SY%
      = 2: GOTO 1090
1050  IF P% = 40 THEN SX% =  - 2:
     SY% = 2: GOTO 1090
1060  IF P% = 8 THEN SX% =  - 2:S
     Y% = 0: GOTO 1090
1070  IF P% = 136 THEN SX% =  - 2
     :SY% =  - 2: GOTO 1090
1080  IF P% = 128 THEN SX% = 0:SY
     % =  - 2: GOTO 1090
1085 AN% = 0:AD = 32768: GOTO 100
     0
1090 AN% = AN% + 1: RETURN
2000  POKE 18706,3: POKE 18707,30
     : POKE 18712,9: POKE 18713,2
     : POKE 18714,4: POKE 18715,9
     : POKE 19328,42
2010  FOR I = 768 TO 773: POKE I,
     0: NEXT
2020  POKE 774,16: POKE 790,16: POKE
     776,12: POKE 794,12
2025  CALL 18691
2030 BN% = 0
2040  RETURN
```

要 RUN 這個程式記着第16行的音樂檔案名稱必須和做好的檔案吻合，筆者的檔案名稱是 IN MAY，而以前所有作爲示範的檔案亦要存在磁碟上。這是一個有畫面，聲音效果和背景音樂的程式，只不過運行較慢而已。如果要較快效果，請把它 COMPILE 爲機械語，則運行時便得到 ARCADE GAME 畫面。

海浪聲和背景音樂的混合

倘若我們希望背景音樂配上鳥鳴和海浪衝擊的效果又如何呢？

海浪衝擊的效果和槍聲有些相似，不同的地方是槍聲一經產生後噪音聲量由最大迅速減弱然後停止，而海浪聲音是由弱至强再由强至弱不斷延續着，讀者可以參考音形（ENVELOP SHAPE）第14種。這種效果的模擬可以利用一個 COUNTER 作爲負責衝擊時間達到，請鍵入下列程式：

```
10  REM  ** PROGRAM TO MIX MUSIC
     & OCEAN EFFECTS **
35  HOME : VTAB 10: INPUT "NAME O
     F PIECE?";A$
40  HOME : PRINT  CHR$ (4);"BLOAD
      MUSIC": PRINT  CHR$ (4);"BL
     OAD ";A$;",A$4000"
100  POKE 18706,3: POKE 18707,30:
      REM  TEMPO & DECAY
110  POKE 18712,9: REM  STRONG
120  POKE 18713,2: POKE 18714,4: REM
      USE CHANNEL C OF 1ST SPEAKE
     R & CHANNEL A OF 2ND SPEAKER

130  POKE 18715,9: REM  REST
```

```
140  POKE 19328,42: REM  NOISE &
     TONE ENABLE
150 R3 = 36897
160 R4 = 36941
180  FOR I = 768 TO 773: POKE I,0
     : NEXT : REM  CLEAR TABLE DA
     TAS
182  POKE 774,16: POKE 790,16: REM
      NOISE PERIOD VALUE
185  POKE 776,12: POKE 794,12: REM
      1ST SPEAKER CHANNEL A AMPLU
     TUDE & 2ND SPEAKER CHANNEL C
      AMPLUTUDE
188  CALL 18691: REM  MUSIC START
      PLAYING
190 N = 0:F = 0: REM  N=BIRD'S PI
     TCH,F=GUNSHOT DURATION
195 T = 1:COUNTER = 0
200  POKE 768,N: POKE 788,N: REM
      INPUT PITCH TO CHANNELS OF
     BOTH SPEAKERS
209  REM  MAKE BIRD'S PITCH RANDO
     M
210 N = N + 1: IF N > 20 THEN N =
      INT ( RND (1) * 10) + 8
220  IF F < 1 THEN T = 1
230  IF F > 10 THEN T =  - 1
240 COUNTER = COUNT + 1
250  IF COUNTER > 2 THEN F = F +
     T:COUNTER = 0
260  REM
300  REM
310  POKE 777,F: POKE 793,F: REM
      INPUT AMPLITUDE OF GUNSHOT
     TO BOTH SPEAKERS
320  GOTO 200
```

我們很容易便把槍聲的程式修改變成海浪衝擊效果。上面第250行的 COUNTER 不斷加一，當它大過指定的數值時 F 的音量值便加以改變，T 值是增强或減弱旗號，當第220行音量值少過一時旗號是正數，表示這時海浪聲由弱至强，而第230行當音量值大過10時則旗號改爲負數，表示海浪聲由强至弱。

把背景音樂配上程式上能使程式有生氣，再加上不同聲音效果，必然更能令程式多姿多彩。不過，各種效果的產生，仍然要靠我們多方面的測試及改進才能達到目的。

第四章

擴大MUSIC CONSTRUCTION SET的範圍

音樂是有生氣而富於感情的藝術，而電腦卻是冰冷的機器，要眞正把兩者連結起來並非容易的事，就某些基本的音樂要求而言，MCS 已經給予我們充份機會去表達，亦可以說，利用 MCS 的確讓我們對音樂有機會加深認識，不過由於電腦記憶體的限制，有很多地方 MCS 卻不能滿足我們的要求。

一隻磁碟可以存放幾多首樂曲？

通常倘若利用 MCS 的 SAVE 指令，我們大概可以存放15首音樂在一隻資料磁碟內。這15首樂曲必須交由 MCS 的MASTER DISK 去演奏，這樣方便我們作出各種修改。不過令人不滿意的地方就是所存樂曲太少了，而且不能讓每一首樂曲反覆不停演奏。存量少的原因最主要是每首樂曲除了音樂資料之外，還有圖形資料的存放，兩種資料一同存放在磁碟上，當然所能夠利用的容量便有所限制了。

反覆依次演奏問題

當 MCS 被 BOOT 起後，假如我們不按 RETURN 的話，它會自動反覆依次序演奏內存的示範樂曲。要它演奏自己所編樂曲，最簡單莫如把自己的樂曲照示範樂曲的檔案名稱 SAVE 到 MCS 內便可以，這時該首樂曲的所有資料便

被我們自己編寫的樂曲資料取代了。

這種方法雖然可以解決問題，但只限於9首，有時我們希望演奏更多樂曲，或編製多首樂曲以供將來程式配樂之用，則需要另外一個方法去處理了。

MUSIC LIST CREATOR

音樂資料既可以被利用作爲背景音樂，又可以加入各種音响效果，當然絕對可以依次序被反覆演奏了。倘若我們只希望踏進一個美妙的電腦音樂世界，認爲毋須顯示音樂符號的圖案，下面所談的一個 MISIC LIST CREATOR 可以給予我們極大的幫助。它可以爲我們存放超過30首以上的樂曲在一隻資料磁碟上，我們可以反覆聆聽其中一首被指定的樂曲，或讓它依次序一一演奏。除非我們向它下命令，否則它會不停演奏，直到永遠！

LISTING　4.1是現在所談的一個程式，應用這個之前，首先把編好的樂曲 BSAVE 到一隻 DATA DISK 上，因爲我們不再需要音樂符號資料，所以只要 BSAVE 的檔案便可以了。每隻 DATA DISK 大概可以存放30首以上樂曲，所有音樂檔案存放好後，取出 DATA DISK，換上 MISIC LIST CREATOR 然後按 RUN 運行該程式，螢幕會印出：

```
INSERT YOUR INITIALIZED DATA DISK



AND PRESS ANY KEY TO MENU...
```

這時可以把磁碟從磁碟機內取出，換回 DATA DISK，然後按任何鍵，電腦跟着會顯示：

```
VALUE OF REST VOLUME

FROM 1 (PIANO) TO 13 (ORGAN)

SELECT REST VOLUME (1 TO 13)?
```

這裡我們需要輸入1至13其中一個值，決定音樂的音色，讓我們鍵入9這個數值，則螢幕會印出一系列的應用功能如下：

```
1. CREATE LISTS OF PIECES TO NEW FILE

2. ADD NAME TO EXISTING LISTS

3. CREATE TEMPO,DECAY TABLES

4. MUSIC MENU

5. PLAY MUSIC OF ALL SELECTED PIECES

6. PLAY MUSIC OF SELECTED PIECE

7. EDIT TEMPO,DECAY TABLES

8. SELECT REST VOLUME
```

第一步要做的是第一項建立一個檔案的列表，將所有樂曲的名稱造成一個表，這個表十分重要，以後所有其他各項應用都需要這個表作爲指引，按1鍵後我們看到

```
OLD MENU WILL BE WRITTEN OVER,
PLEASE CREATE A NEW TEMPO TABLES.

ARE YOU READY?(Y/N)
```

表示如果 DATA DISK 已經有一個列表的話，則新的列表會把舊的一個取代了，而新的列表要一個新的 TEMPO TABLES（下面會談到），由於我們的 DATA DISK 還沒有任何列表，所以按 Y 鍵開始建立列表

```
PRESS KEY 'D' TO STORE LIST ON DISK
PRESS ANY OTHER KEY TO EXIST
```

跟着按 D 鍵，電腦會打出

```
ENTER NAME OF PIECE;TYPE ! TO STOP
```

告訴我們把樂曲檔案名稱逐一鍵入，直至完畢後，按！鍵，則一個樂曲列表便交由電腦編製出來了。

列表本身是一個樂曲的名單，當電腦編製完畢後，螢幕會返回應用目錄，這時按4鍵便會看到一個樂曲的名單了，筆者預先已經編好一些示範的樂曲在一隻 DATA DISK 內，倘若按4鍵，則會有如下顯示：

```
LOADING MENU DATA...



END OF DATA

PRESS ANY KEY TO CONTINUE
```

以上的顯示當我們按任何目錄上的應用項目都有機會出現，表示電腦從列表內輸入資料，最後告訴我們資料輸入完畢，可以按任何鍵顯示樂曲名單了。

從示範 DATA DISK 裡，我們清楚地看到一個如下的名單：

```
1       GOLDFINGER
2       GODFATHER
3       NO MORE
4       YOU ONLY LIVE TWICE
5       LOVE STORY
6       FORM RUSSIA WITH LOVE
7       SOUND OF SLIENCE
8       LOVE IS BLUE
9       WE MAY NEVER LOVE THIS AGAIN
10      THE MAGNIFICENT SEVEN
11      I DON'T KNOW HOW TO LOVE HIM
12      TARA'S THEME
13      YESTERDAY
14      ENCHANTED EVENING
15      THE WAY WE WERE


PRESS A KEY TO MENU..
```

樂曲依次序排列起來，我們現時知道究竟 DATA DISK 內有些甚麼樂曲了，這時可按任何鍵返回目錄。

樂曲有了名稱列表，亦需要一個速度和樂曲消逝的列表，因爲每首樂曲速度不同，音色亦有異，所以必須按3鍵再建立一個速度資料列表，首先電腦會詢問我們是否有表存在，如果已經有的話，按 Y 鍵去改變該表資料，相反來說按 N 鍵去建立一個新表，螢幕跟着會把樂曲名稱依次序逐一顯示，讓我們輸入速度和消逝的數值，舉例如下：

```
DO YOU HAVE EXISTING TABLES?(Y/N)

1 GOLDFINGER

TEMPO (FAST 1 TO SLOWER 30)3

DECAY (1 TO 50)5


2 GODFATHER

TEMPO (FAST 1 TO SLOWER 30)4

DECAY (1 TO 50)10


3 NO MORE

TEMPO (FAST 1 TO SLOWER 30)4

DECAY (1 TO 50)20
```

當樂曲列表內所有速度和消逝資料被輸入後，電腦便會自動替我們建立一個速度列表了。我們無須担心是否輸入所有樂曲的資料，當樂曲名稱表內依次序被輸入後，電腦立即會打出 TABLE CREATION IN PROCESS。

假如我們完成以上所有程序，我們想加入新的樂曲，或過了數日後我們又編製了其他樂曲，目錄檔案第2項讓我們可以把新曲加入，按2鍵我們得到：

```
PIECE TO BE ADD FROM NO.15
PRESS KEY 'A' TO ADD NEW PIECE
PRESS ANY OTHER KEY TO EXIT
```

跟着按 A 鍵讓我們可以依次序加入新的樂曲。

樂曲一經加入，當然亦要加入新的速度和消逝資料。事實上，我們亦可以隨時更改每首樂曲的速度和消逝資料，返回目錄內按7鍵，電腦會把所有樂曲顯示出來：

```
END OF DATA

PRESS ANY KEY TO CONTINUE

1     GOLDFINGER
2     GODFATHER
3     NO MORE
4     YOU ONLY LIVE TWICE
5     LOVE STORY
6     FORM RUSSIA WITH LOVE
7     SOUND OF SLIENCE
8     LOVE IS BLUE
9     WE MAY NEVER LOVE THIS AGAIN
10    THE MAGNIFICENT SEVEN
11    I DON'T KNOW HOW TO LOVE HIM
12    TARA'S THEME
13    YESTERDAY
14    ENCHANTED EVENING
15    THE WAY WE WERE
 PLEASE SELECT PIECE NO. TO BE EDIT..15
```

並且在畫面下面告訴我們選擇需要加入或更改資料的樂曲編號，輸入編號後，則電腦會顯示 LOADING TEMPO DATA，讓我們作出速度和消逝值的輸入。

所有資料編製完畢，可以聆聽樂曲了。目錄第5項電腦會依次序把樂曲演奏，而第6項則只演奏指定的樂曲，至於第8項讓我們隨時改變演奏的 REST 值。

```
1  REM *************************
2  REM * MUSIC LIST CREATOR    *
3  REM *************************
4  HOME
5  DIM MN$(50)
10  NU  =  0:TEMPO = 18706:DECAY  =  18707:REST  =
18715:TA = 20480:DA = 20530:RA = 20580
15 D$ =  CHR$ (4): PRINT D$;"BLOAD MUSIC"
17  GOSUB 6500
18   HOME :  VTAB 10:  HTAB 2:  PRINT "INSERT YOUR
INITIALIZED DATA DISK":  VTAB 14:  HTAB  4:  PRINT
"AND PRESS ANY KEY TO MENU...": POKE  - 16368,0
19  IF  PEEK ( - 16384) < 128 THEN 19
20  HOME :  VTAB 10:  HTAB 2: PRINT "VALUE OF REST
VOLUME":  PRINT : PRINT : PRINT "FROM 1 (PIANO) TO
13 (ORGAN)": PRINT : PRINT
22   INPUT "SELECT REST VOLUME (1 TO 13)?";SN:  IF
SN < 0 OR SN > 13 THEN 22
24  POKE 18715,SN
25  HOME
30  HTAB 2:  PRINT "1.  CREATE LISTS OF PIECES  TO
NEW FILE": PRINT
40  HTAB 2: PRINT "2. ADD NAME TO EXISTING LISTS":
PRINT
50  HTAB 2:  PRINT "3. CREATE TEMPO,DECAY TABLES":
PRINT
60  HTAB 2: PRINT "4. MUSIC MENU": PRINT
65  HTAB 2:  PRINT "5.  PLAY MUSIC OF ALL PIECES":
PRINT
67   HTAB  2:  PRINT "6.  PLAY MUSIC  OF  SELECTED
PIECE": PRINT
68   HTAB 2:  PRINT "7.  EDIT TEMPO,DECAY TABLES":
PRINT
69  HTAB 2: PRINT "8. SELECT REST VOLUME": PRINT
70  GOSUB 200
72  IF I = 8 THEN 20
75  IF I = 7 THEN  GOSUB 4000: GOTO 25
78  IF I = 6 THEN 3000
80  IF I = 5 THEN  GOSUB 7000: HOME : GOTO 7100
82  IF I = 4 THEN NU = 0:  GOSUB 7000: GOSUB 7055:
VTAB 24:  PRINT "PRESS A KEY TO MENU..";:  GET B$:
GOTO 25
85  ON I GOSUB 1000,2000,5000
90  GOTO 25
200  REM
```

```
210  INVERSE :  PRINT "SELECT A NUMBER";: NORMAL :
PRINT " ";
220  GET A$
230 I =  VAL (A$)
240  IF I < 1 OR I > 8 THEN 220
250  RETURN
1000  REM
1010  GOSUB 1200
1020  HOME : PRINT "PRESS KEY 'D' TO STORE LIST ON
DISK"
1030  PRINT "PRESS ANY OTHER KEY TO EXIT"
1040  GET A$
1050  IF A$ < > "D" THEN  RETURN
1060  PRINT  CHR$ (13)
1070  PRINT D$;"OPEN DISK"
1080  PRINT D$;"DELETE DISK"
1090  PRINT D$;"OPEN DISK"
1100  GOSUB 1300
1110  PRINT D$;"CLOSE DISK"
1120  RETURN
1200  HOME :  PRINT  CHR$ (7):  VTAB 5: PRINT "OLD
MENU WILL BE WRITTEN OVER,"
1210  PRINT "PLEASE CREATE A NEW TEMPO TABLES."
1220  REM
1230   PRINT  :  PRINT  :  PRINT  "ARE  YOU
READY?(Y/N)";: GET C$
1240  IF C$ < > "Y" THEN  POP : GOTO 25
1250  RETURN
1300  REM
1310  PRINT "ENTER NAME OF PIECE; TYPE ! TO STOP"
1320  INPUT W$
1330  IF W$ = "!" THEN  RETURN
1340  PRINT D$;"WRITE DISK"
1350  PRINT W$
1355  PRINT D$
1360  GOTO 1320
2000 NU = 0: GOSUB 7000:NN = NU
2010  HOME : PRINT "PIECE TO BE ADD FROM NO.";NN
2020  PRINT "PRESS KEY 'A' TO ADD NEW PIECE"
2030  PRINT "PRESS ANY OTHER KEY TO EXIT"
2040  GET A$
2050  IF A$ < > "A" THEN  RETURN
2060  PRINT  CHR$ (13)
2070  PRINT D$;"APPEND DISK"
2080  GOSUB 1300
```

```
2090  PRINT D$;"CLOSE DISK"
2100 NU = 0: GOSUB 7000
2110  PRINT  CHR$ (13)
2120  PRINT D$;"BLOAD DISK.TEMPO"
2125  PRINT D$;"BLOAD DISK.DECAY"
2130  HOME
2140  FOR P = NN + 1 TO NU
2150  PRINT P;"  ";MN$(P): PRINT
2160  INPUT "TEMPO (FAST 1 TO SLOW 30)";T
2170  POKE TA + P,T
2220  PRINT : INPUT "DECAY (1 TO 50)";D
2225  POKE DA + P,D
2230  NEXT P
2240  GOSUB 5140
2250  RETURN
3000  GOSUB 7000: GOSUB 7055
3010  VTAB 24:  HTAB 2: INPUT "PLEASE SELECT PIECE
NO..."; PN
3020   IF PN > NU THEN  VTAB  24:  HTAB  2:  PRINT
"PIECE NO.  NOT EXISTS IN MENU.....": FOR J = 1 TO
100: NEXT : GOTO 3010
3030  HOME : VTAB 10: HTAB 2: PRINT "LOADING TEMPO
DATA...."
3040   PRINT   CHR$   (13):   PRINT   D$;"BLOAD
DISK.TEMPO"
3045  PRINT D$;"BLOAD DISK.DECAY"
3050  FOR K = 1 TO 100: NEXT
3060 L = PN: GOSUB 10000
3070  POKE  - 16368,0
3080  CALL 18691
3090   IF  PEEK ( - 16384) > 127 THEN  CALL 18700:
GOTO 25
3100  GOTO 3090
4000  GOSUB 7000: HOME : GOSUB 7055
4010  VTAB 24:  HTAB 2:  INPUT "PLESE SLECT  PIECE
NO. TO BE EDIT..";PN
4020   IF  PN > NU THEN  VTAB 24:  HTAB  2:  PRINT
"PIECE NO.  NOT EXISTS IN MENU.....": FOR J = 1 TO
100: NEXT : GOTO 4010
4030  HOME :  VTAB 10: HTAB 2: PRINT "LOADING TEMP
DATA....."
4040   PRINT  CHR$ (13):  PRINT  CHR$  (4);"BLOAD
DISK.TEMPO": PRINT  CHR$ (4);"BLOAD DISK.DECAY"
4050  HOME : PRINT PN;"  ";MN$(PN): PRINT
4060  INPUT "TEMPO (FAST 1 TO SLOW 30)";T
```

```
4110   POKE TA + PN,T
4115   PRINT : INPUT "DECAY (1 TO 50)";D
4116   POKE DA + PN,D
4120   GOTO 5140
5000   HOME : GOSUB 7000: HOME
5010   VTAB 4:  HTAB 2: PRINT "DO YOU HAVE EXISTING
TABLES?(Y/N)";: GET A$
5020   IF A$ <  > "Y" THEN 5040
5025   PRINT  CHR$ (13)
5030   PRINT D$;"BLOAD DISK.TEMPO"
5040   HOME
5050   FOR I = 1 TO NU
5060   PRINT I;"  ";MN$(I): PRINT
5070   INPUT "TEMPO (FAST 1 TO SLOWER 30)";T
5080   POKE TA + I,T
5125   PRINT : INPUT "DECAY (1 TO 50)";D
5128   POKE DA + I,D: PRINT : PRINT
5130   NEXT I
5140    HOME  :  VTAB  10:  HTAB  2:  PRINT  "TABLE
CREATION IS IN PROCESS.."
5150   PRINT D$;"BSAVE DISK.TEMPO,A20480,L50"
5160   PRINT D$;"BSAVE DISK.DECAY,A20530,L50"
5170   RETURN
6000   REM
6003   CALL 19968
6005 EC =  PEEK (222)
6010    IF EC = 5 THEN  PRINT "END OF  DATA":  GOTO
       6050
6020   PRINT "ERROR NO. ";EC
6050   PRINT D$;"CLOSE DISK"
6060   PRINT :  PRINT "PRESS ANY KEY TO CONTINUE";:
GET A$
6070   RETURN
6100   REM
6105   CALL 19968
6110 EC =  PEEK (222)
6120   IF EC = 5 THEN 3340
6130   PRINT "ERROR NO. ";EC
6140   GOTO 3420
6500   REM
6510   DATA  104,168,104,166,223,154,72,152,72,96
6520   FOR A = 19968 TO 19977
6530   READ C
6540   POKE A,C
6550   NEXT A
```

```
6560  RETURN
7000  NU  = 0:  HOME :  VTAB  10:  HTAB  2:  PRINT
"LOADING MENU DATA...": ONERR  GOTO 6000
7005  PRINT , CHR$ (13)
7010  PRINT D$;"OPEN DISK"
7020  PRINT D$;"READ DISK"
7030  INPUT W$
7040 NU = NU + 1:MN$(NU) = W$
7050  GOTO 7030
7055  HOME
7060  FOR I = 1 TO NU
7070  PRINT I;: HTAB 6: PRINT MN$(I)
7080  IF I / 20 =  INT (I / 20) THEN  PRINT "PRESS
ANY KEY TO CONTINUE..": GET A$
7090  NEXT
7095  IF MN$(1) = "" THEN  HOME : VTAB 10: HTAB 2:
PRINT "MENU NOT EXISTS!!":  PRINT :  PRINT "PLEASE
CREATE A MENU."
7098  RETURN
7100  VTAB 24:  HTAB 2: PRINT "DO YOU WANT TO PLAY
THE MUSIC?(Y/N)";: GET A$
7200  IF A$ <  > "Y" THEN 25
7210  HOME : VTAB 10: HTAB 2: PRINT "LOADING TEMPO
DATA...."
7220   PRINT   CHR$   (13):   PRINT   D$;"BLOAD
DISK.TEMPO": PRINT D$;"BLOAD DISK.DECAY"
7225  POKE  - 16368,0
7230 L = 1
7240  GOSUB 10000
7245  FOR I = 768 TO 799: POKE I,0: NEXT
7250  CALL 18691
7260  FOR J = 1 TO 700: NEXT
7270   IF  PEEK (6) > 253 AND  PEEK (7) < 64  THEN
CALL 18700: GOTO 7290
7275   IF  PEEK ( - 16384) > 127 THEN  CALL 18700:
GOTO 25
7280  GOTO 7270
7290 L = L + 1:  IF L > NU THEN L = 1
7300  GOTO 7240
10000  HOME :  PRINT  CHR$ (13):  VTAB 10: HTAB 2:
PRINT L;"  ";MN$(L):  VTAB 24:  PRINT "PRESS A KEY
TO MENU..": PRINT D$;"BLOAD ";MN$(L);",A$4000"
10010  POKE TEMPO, PEEK (TA + L): POKE DECAY, PEEK
(DA + L)
10020  RETURN
```

演奏更長的樂曲

MCS 所供給我們演奏的樂曲長度有所限制，大約有70個小節左右。對於較長的樂曲無法演奏，有些樂曲分爲若干樂章，每一個樂章會有不同速度去表達，例如一些奏鳴曲或舞曲之類，包括有慢板，快板在內，要演奏這些樂曲，需要作某些程度的安排。

LISTING 4.2是一個 LONG MUSIC CREATOR。顧名思義讓我們演奏這些較長的樂曲，現在讓我們研究一下如何運用。

原則上，既然我們可以把每一首樂曲依次序演奏，順理成章亦可以把幾首樂曲合而爲一，問題是必須同時把幾首樂曲的資料存入電腦的記憶體內，這樣毋須一個樂章完畢後再從磁碟上輸入資料。以48K 的電腦來說，足夠我們同時存放5首樂曲的資料，換言之，我們可以一次演奏一首 MCS 所給予我們音樂長度5倍之多。

在未應用 LONG MUSIC CREATOR 之前，先把音樂資料存放在 DATA DISK 內。我們要另外編製一隻 DATA DISK。首先把每一首樂曲分爲數段，最多可以分爲5段，視乎樂曲長度而定。倘若每一段的節奏快慢不同，則要作出適當的分段，然後每段分別 BSAVE 到 DATA DISK 內，例如筆者在示範磁碟內一首貝多芬的小步舞曲分爲 MINUET 1-MINUET 2和 MINUET 3三個分段檔案。每個分段檔案我們稱之爲 PHASE。由於分段檔案都是同一首樂曲的一部份，所以最好用同一樂曲名稱而後面加上數目加以分辨。

```
1  REM  *************************
2  REM  * LONG MUSIC CREATOR    *
3  REM  *************************
4  HOME
5  DIM MN$(10),MD$(5)
10  NU = 0:TEMPO = 18706:DECAY = 18707:TA =
20480:MOVE = 20224
15 D$ = CHR$ (4): PRINT D$;"BLOAD MUSIC"
16  PRINT D$;"BLOAD MOVE"
17  GOSUB 6500
18  HOME : VTAB 10: HTAB 2: PRINT "INSERT YOUR
INITIALIZED DATA DISK": VTAB 14: HTAB 4: PRINT
"AND PRESS ANY KEY TO MENU...": POKE - 16368,0
19  IF PEEK ( - 16384) < 128 THEN 19
20  HOME : VTAB 10: HTAB 2: PRINT "VALUE OF REST
VOLUME": PRINT : PRINT : PRINT "FROM 1 (PIANO TO
13 (ORGAN)": PRINT
22  INPUT "SELECT VALUE (1 TO 13)?";SN: IF SN < 0
OR SN > 13 THEN 22
24  POKE 18715,SN
25  HOME
30  HTAB 2: PRINT "1. CREATE LISTS OF PIECES TO
NEW FILE": PRINT
40  HTAB 2: PRINT "2. ADD NAME TO EXISTING LISTS":
PRINT
50  HTAB 2: PRINT "3. CREATE AND EDIT TEMPO
TABLES": PRINT
60  HTAB 2: PRINT "4. MUSIC MENU": PRINT
65  HTAB 2: PRINT "5. PLAY MUSIC OF ALL PIECES":
PRINT
67  HTAB 2: PRINT "6. PLAY MUSIC OF SELECTED
PIECE": PRINT
68  HTAB 2: PRINT "7. CREATE DIFFERENT PHASE OF
EACH PIECE": PRINT
69  HTAB 2: PRINT "8. SELECT REST VOLUME": PRINT
70  GOSUB 200
72  IF I = 8 THEN 20
75  IF I = 7 THEN GOSUB 8000: GOTO 25
78  IF I = 6 THEN 3000
80  IF I = 5 THEN GOSUB 7000: GOSUB 7055: GOTO
7100
82  IF I = 4 THEN NU = 0: GOSUB 7000: GOSUB 7055:
VTAB 24: PRINT "PRESS A KEY TO MENU..";: GET B$:
```

```
GOTO 25
85  ON I GOSUB 1000,2000,5000
90  GOTO 25
200  REM
210  INVERSE :  PRINT "SELECT A NUMBER";: NORMAL :
PRINT " ";
220  GET A$
230 I =  VAL (A$)
240  IF I < 1 OR I > 8 THEN 220
250  RETURN
1000  REM
1010  GOSUB 1200
1020  HOME : PRINT "PRESS KEY 'D' TO STORE LIST ON
DISK"
1030  PRINT "PRESS ANY OTHER KEY TO EXIT"
1040  GET A$
1050  IF A$ <  > "D" THEN  RETURN
1060  PRINT  CHR$ (13)
1070  PRINT D$;"OPEN DISK"
1080  PRINT D$;"DELETE DISK"
1090  PRINT D$;"OPEN DISK"
1100  GOSUB 1300
1110  PRINT D$;"CLOSE DISK"
1120  RETURN
1200  HOME :  PRINT  CHR$ (7):  VTAB 5: PRINT "OLD
MENU WILL BE WRITTEN OVER,"
1210  PRINT "PLEASE CREATE A NEW TEMPO TABLES."
1215  PRINT : PRINT "MAXIMUM NO. OF PIECES IS 7"
1220  REM
1230    PRINT   :   PRINT   :   PRINT   "ARE   YOU
READY?(Y/N)";: GET C$
1240  IF C$ <  > "Y" THEN  POP : GOTO 25
1250  RETURN
1300 NV = 0
1310  PRINT "ENTER NAME OF PIECE; TYPE ! TO STOP"
1320 NV = NV + 1: INPUT W$
1330  IF W$ = "!" OR NV > 10 THEN  RETURN
1340  PRINT D$;"WRITE DISK"
1350  PRINT W$:MN$(NV) = W$
1355  PRINT D$
1360  GOTO 1320
2000 NU = 1: GOSUB 7000:NN = NU
2010  HOME : PRINT "PIECE TO BE ADD FROM NO.";NN
```

```
2020  PRINT "PRESS KEY 'A' TO ADD NEW PIECE"
2030  PRINT "PRESS ANY OTHER KEY TO EXIT"
2040  GET A$
2050  IF A$ <  > "A" THEN  RETURN
2060  PRINT  CHR$ (13)
2070  PRINT D$;"APPEND DISK"
2080  GOSUB 1300
2090  PRINT D$;"CLOSE DISK"
2100  GOTO 25
3000  GOSUB 7000: GOSUB 7055
3010  VTAB 24:  HTAB 2: INPUT "PLEASE SELECT PIECE
NO...";PN
3020   IF  PN > NU THEN  VTAB 24:  HTAB  2:  PRINT
"PIECE NO.  NOT EXISTS IN MENU.....": FOR J = 1 TO
100: NEXT : GOTO 3010
3030  HOME : VTAB 10: HTAB 2: PRINT "LOADING TEMPO
DATA...."
3040   PRINT   CHR$   (13):   PRINT   D$;"BLOAD
";MN$(PN);".TEMPO"
3050  GOSUB 9000
3060  POKE  - 16368,0
3070 L = 1:ML = 24576
3080  GOSUB 10000
3090 L = L + 1:ML = ML + 2305: IF L > NU THEN 3120
3100  GOTO 3080
3120  HOME :  VTAB 10: HTAB 4: PRINT MN$(PN): VTAB
24: HTAB 4: PRINT "PRESS ANY KEY TO MENU"
3130 L = 1:ML = 24576:ME = 26879:MD = 16384
3140  POKE TEMPO,  PEEK (TA + L): POKE DECAY, PEEK
(TA + L): POKE 255,L - 1: CALL 20224
3150  CALL 18691
3160  FOR J = 1 TO 700: NEXT
3170   IF  PEEK (6) > 253 AND  PEEK (7) < 64  THEN
3200
3180  IF  PEEK ( - 16384) > 127 THEN  CALL  18700:
GOTO 25
3190  GOTO 3170
3200  CALL 18700:L = L + 1: IF L > NU THEN 3220
3210  GOTO 3140
3220  CALL 18700
3230  FOR J = 1 TO 100: NEXT
3240  GOTO 3130
5000  GOSUB 7000: GOSUB 7055
```

```
5010  VTAB 24:  HTAB 2: INPUT "PLEASE SELECT PIECE
NO..";PN
5015  IF PN > NU THEN VTAB 24: HTAB 2: PRINT
"PIECE NO. NOT EXISTS IN MENU..": FOR J = 1 TO
100: NEXT : GOTO 5010
5020  HOME
5030  GOSUB 9000
5040  HOME
5050  FOR I = 1 TO NU
5060  PRINT I;"   ";MD$(I): PRINT
5070  INPUT "TEMPO (FAST 1 TO SLOWER 30)";T
5080  POKE TA + I,T
5125  PRINT : PRINT
5130  NEXT I
5140  HOME : VTAB 10: HTAB 2: PRINT "TABLE
CREATION IS IN PROCESS.."
5150                PRINT                D$;"BSAVE
";MN$(PN);".TEMPO,A20480,L10"
5160  RETURN
6000  REM
6003  CALL 19968
6005 EC =  PEEK (222)
6010  IF EC = 5 THEN  PRINT "END OF  DATA":  GOTO
6050
6020  PRINT "ERROR NO. ";EC
6050  PRINT D$;"CLOSE DISK"
6060  PRINT :  PRINT "PRESS ANY KEY TO CONTINUE";:
GET A$
6070  RETURN
6100  REM
6105  CALL 19968
6110 EC =  PEEK (222)
6120  IF EC = 5 THEN 3340
6130  PRINT "ERROR NO. ";EC
6140  GOTO 3420
6500  REM
6510  DATA  104,168,104,166,223,154,72,152,72,96
6520  FOR A = 19968 TO 19977
6530  READ C
6540  POKE A,C
6550  NEXT A
6560  RETURN
6600  CALL 19968:EC =  PEEK (222)
```

```
6610  IF EC = 5 THEN  PRINT D$;"CLOSE DISK"
6620  RETURN
7000  NU  = 0:  HOME :  VTAB  10:  HTAB  2:  PRINT
"LOADING MENU DATA...": ONERR  GOTO 6000
7005  PRINT  CHR$ (13)
7010  PRINT D$;"OPEN DISK"
7020  PRINT D$;"READ DISK"
7030  INPUT W$
7040 NU = NU + 1:MN$(NU) = W$
7050  GOTO 7030
7055  HOME
7060  FOR I = 1 TO NU
7070  PRINT I;: HTAB 6: PRINT MN$(I)
7080  REM
7090  NEXT
7092 NN = NU
7095  IF MN$(1) = "" THEN  HOME : VTAB 10: HTAB 2:
PRINT "MENU NOT EXISTS!!":  PRINT :  PRINT "PLEASE
CREATE A MENU."
7098  RETURN
7100  VTAB 24:  HTAB 2: PRINT "DO YOU WANT TO PLAY
THE MUSIC?(Y/N)";: GET A$
7200  IF A$ <  > "Y" THEN 25
7210 PN = 1
7220   GOSUB 9100:  GOSUB 9000:  FOR J = 1 TO 200:
NEXT
7225  POKE  - 16368,0
7230 L = 1:ML = 24576
7240  GOSUB 10000
7290 L = L + 1:ML = ML + 2305:  IF L > NU THEN 7400
7300  GOTO 7240
7400  HOME :  VTAB 10: HTAB 4: PRINT MN$(PN): VTAB
24: HTAB 4: PRINT "PRESS ANY KEY TO MENU"
7500 L = 1:ML = 24576:ME = 26879:MD = 16384
7510  POKE TEMPO,  PEEK (TA + L): POKE DECAY, PEEK
(TA + L): POKE 255,L - 1: CALL 20224
7520  CALL 18691
7530  FOR J = 1 TO 700: NEXT
7540   IF  PEEK (6) > 253 AND  PEEK (7) < 64  THEN
7600
7545  IF  PEEK ( - 16384) > 127 THEN  CALL  18700:
GOTO 25
```

```
7550  GOTO 7540
7600  CALL 18700:L = L + 1: IF L > NU THEN 7700
7610  GOTO 7510
7700  CALL 18700
7710 PN = PN + 1: IF PN > NN THEN PN = 1
7720  GOTO 7220
8000  GOSUB 7000: GOSUB 7055
8010  VTAB 24:  HTAB 2: INPUT "PLEASE SELECT PIECE
NO..";PN
8020  IF  PN > NU THEN  VTAB 24:  HTAB  2:  PRINT
"PIECE  NO.  NOT EXISTS IN MENU..":  FOR J = 1  TO
100: NEXT : GOTO 8010
8030  PRINT  CHR$ (13)
8040  PRINT D$;"OPEN ";MN$(PN);".PH"
8050  PRINT D$;"DELETE ";MN$(PN);".PH"
8060  PRINT D$;"OPEN ";MN$(PN);".PH"
8064  GOSUB 8070
8065  PRINT D$;"CLOSE ";MN$(PN);".PH"
8068  RETURN
8070 NW = 0:  PRINT "ENTER NAME OF PHASE;TYPE ! TO
STOP"
8080  INPUT W$
8090  NW  = NW + 1:  IF W$ = "!" OR NW  >  5  THEN
RETURN
8100  PRINT D$;"WRITE ";MN$(PN);".PH"
8110  PRINT W$:MD$(NW) = W$
8120  PRINT D$
8130  GOTO 8080
9000  NU  = 0:  HOME :  VTAB  10:  HTAB  2:  PRINT
"LOADING PHASE DATA..": ONERR  GOTO 6600
9010    PRINT    CHR$   (13):    PRINT   D$;"OPEN
";MN$(PN);".PH"
9020  PRINT D$;"READ ";MN$(PN);".PH"
9030  INPUT W$
9040 NU = NU + 1:MD$(NU) = W$
9050  GOTO 9030
9100  HOME :  PRINT  CHR$ (13):  PRINT  D$;"BLOAD
";MN$(PN);".TEMPO": RETURN
10000  HOME :  PRINT  CHR$ (13):  PRINT  D$;"BLOAD
";MD$(L);",A";ML: RETURN
```

LONG MUSIC CREATOR 需要一個把音樂資料移動的副程式把資料在記憶體內移動，下面是一個移動的機械碼，鍵入後請以 MOVE 名稱 BSAVE 到存有 LONG MUSIC CREATOR 的磁碟上。起始位置是＄4F00，而長度是＄3C。

```
4F00- A6 FF BD 24 4F 85 3D BD
4F08- 2A 4F 85 3C A9 40 85 43
4F10- A9 00 85 42 BD 30 4F 85
4F18- 3F BD 36 4F 85 3E A0 00
4F20- 20 2C FE 60 60 69 72 7B
4F28- 84 8D 00 01 02 03 04 05
4F30- 68 72 7B 84 8D 96 FF 00
4F38- 01 02 03 04 FF
```

在這裡要提示一下就是無論那一個 CREATOR 都需要上一章提過的 MUSIC 同時存在磁碟上（不是 DATA DISK）讓 CREATOR 加以運用，否則不會產生音樂。

現在就 RUN 這個 LONG MUSIC CREATOR 看看它做些甚麼工作。所有功能和上面的 MISIC LIST CREATOR 一樣，不過目錄會有如下顯示：

```
1. CREATE LISTS OF PIECES TO NEW FILE
2. ADD NAME TO EXISTING LISTS
3. CREATE AND EDIT TEMPO TABLES
4. MUSIC MENU
5. PLAY MUSIC OF ALL PIECES
6. PLAY MUSIC OF SELECTED PIECE
7. CREATE DIFFERENT PHASE OF EACH PIECE
8. SELECT REST VOLUME
```

首先第1項是樂曲列表，樂曲名稱同樣依次序輸入，雖然每一首樂曲包括數個 PHASE，但名稱事實上只有一個，所以把每一首樂曲名稱輸入便可，無須加上 PHASE 的編號，電腦演奏時會自行決定那一段演奏。每隻磁碟可以同時存放大約7至8首擁有5個 PHASE 的樂曲。

造好樂曲列表後，螢幕會返回目錄，這時需按7鍵去建立 PHASE 列表。每一首樂曲由不同段落組成，演奏 PHASE 的次序由這個列表決定。當電腦運行之後，畫面上會把所有樂曲名稱加上編號排列出來，例如：

```
1     MINUET IN G
2     CAMP OF THE GYPSIES
3     SINGING BOAT
4     JOY OF SPRING
5     TRAUMEREI
6     SUMMER NIGHT
```

而畫面下面則會告訴我們選擇那一首樂曲去建立 PHASE 表

```
PLEASE SELECT PIECE NO..
```

鍵入所需編號後，電腦跟着會印出 ENTER NAME OF PHASE，TYPE ！TO STOP 表示我們現在可以依次序把每一個 PHASE 的檔案名稱輸入。名稱必須和 BSAVE 到磁碟一樣，否則電腦找不到應該存入的檔案。

下一步要做的就是輸入速度資料，一首樂曲的每個分段可能有不同節奏，目錄第3項便是讓我們逐一把每一個 PHASE 的速度輸入，按3鍵然後照 MISIC LIST CREATOR 方法一樣輸入速度數值，則電腦便會替我們建立一個速度資料檔案。

以上兩個音樂擴展程式爲 MUSIC CONSTRUCTION SET 而設，可以使我們更加充份發揮該軟件的音樂功能，亦進入更佳的電腦音樂領域。

第五章

把聲音和音樂加入程式實例

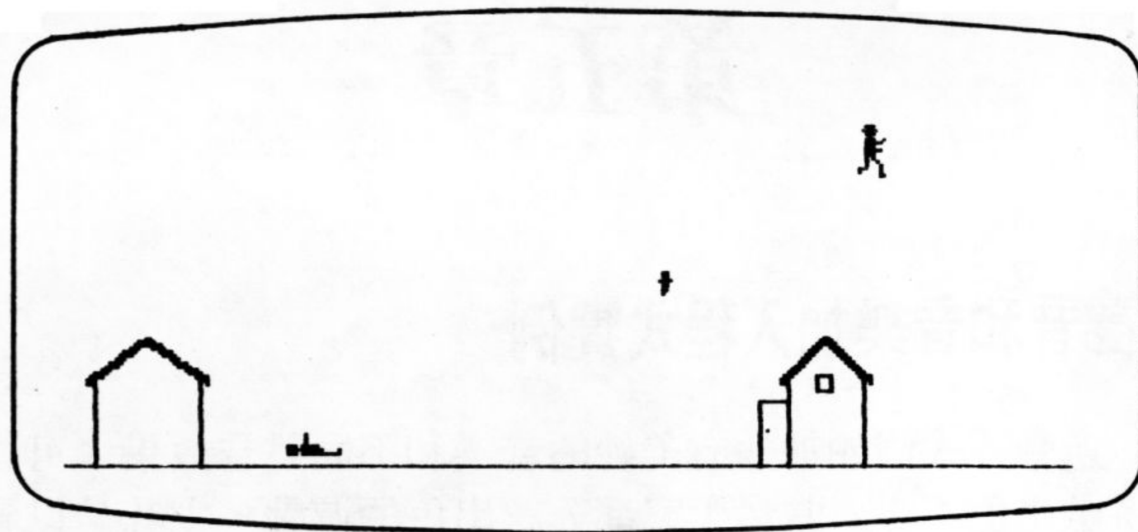
遊戲程式假如加插有美妙的音樂和不同聲音效果，可以使遊戲生色不少。當然大家都公認出色的遊戲，尤其是以動畫爲主的大部份都由組合語言所編製。故此對由 BASIC 這種高階層語言所編寫的程式比較得到較低評價。

事實上，對電腦發生興趣大多由 GAME 開始，而學習電腦語言除非有特殊需要，亦由 BASIC 開始，BASIC 運行雖然緩慢，但正好供給我們一個學習語言的好機會，因爲無論編輯、除錯、修改都十分容易，電腦時代一直以來都有很多這類 BASIC 編寫的遊戲程式刊出，其中不乏出色的創作，筆者在這一章把其中三個加上以往幾章的技巧，爲這些遊戲配上一套新的衣服。

(一)高空擲物 DANGER ROAD

一個原載於第9期的遊戲，作者把一個咭片機遊戲生動地搬到 APPLE 讓我們可以在電腦上進行。程式設計十分精簡，我們很容易便可以爲它配上背景音樂和聲音效果。

第5行我們首先把 MUSIC 和一首預備好的音樂資料存入記憶體內。IN MAY 是筆者自己編的音樂資料檔案，所有以下的程式大家都可以隨意把檔案名稱改爲你自己喜愛的音樂曲調。因爲程式不算長而且第二頁高解像並未有用到，



所以 MUSIC 和音樂資料都同時放入第4頁記憶體內，（MUSIC 程式在 MUSIC CONSTRUCTION 內是由 $4900 開始）

跟着6000行的副程式是輸入音樂速度，消逝和所需聲道資料，並且把第三頁 MCS 和 MOCKINGBOARD 溝通的記憶暫時清除並設定爲零。這一步十分重要，避免有其他雜聲出現。

由於應用 MOCKINGBOARD 的關係，原有觸動 APPLE 喇叭的部份程式都刪去了。

第400行呼叫 MUSIC 產生背景音樂。

原有程式產生聲音效果是呼叫2000行的副程式，所以由2000行開始，把聲音效果改爲由 MOCKINGBOARD 發聲。這裡 DE 這個變數是音量值，把 DE 改變可產生不同的聲响。由2000至2130行都是幾種聲音效果的副程式。這些效果如何編製在第三章已經有詳細說明。

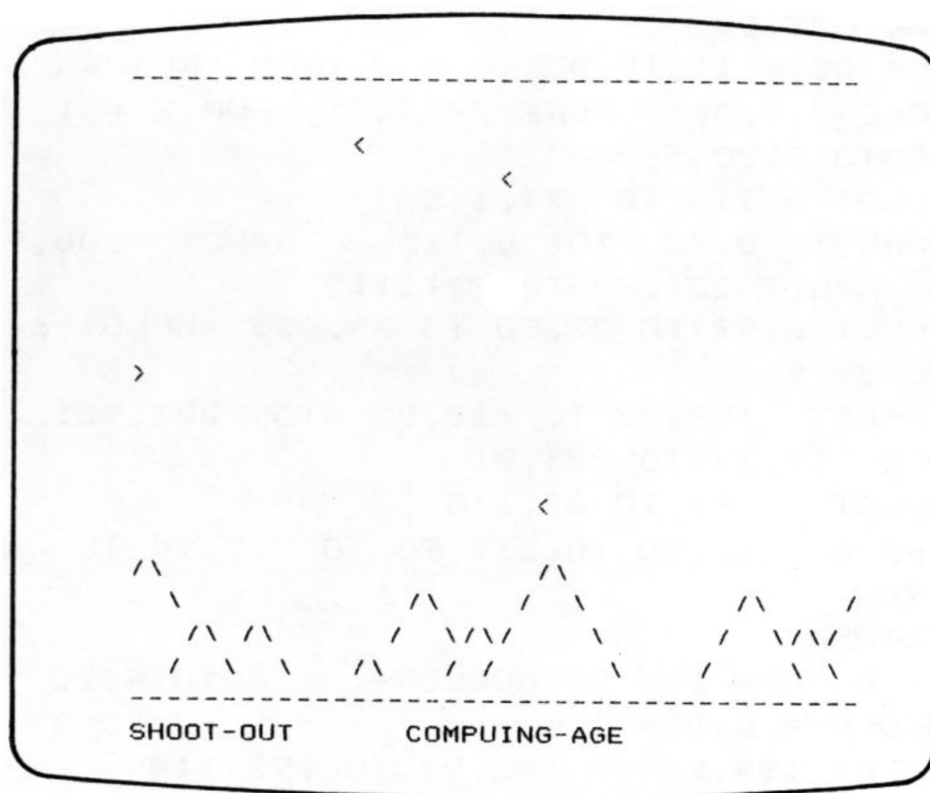
第4060行是當遊戲完結時，把聲响聲道全部關閉，再停止背景音樂，然後才詢問玩者是否再繼續遊戲。

```
1  REM  *************************
2  REM  *  DANGER      ROAD     *
3  REM  *************************
5  D$  =  CHR$ (13) +  CHR$ (4):  PRINT  D$;"BLOAD
MUSIC": PRINT D$;"BLOAD IN MAY,A$4000"
8  GOSUB 6000
10  HIMEM: 36860: HOME : VTAB 23: PRINT "SCORE :";
20 D$ = "100000":R = 36864:W =  - 16336:D = 1
30  HGR : SCALE= 1: HCOLOR= 3: ROT= 0: GOSUB 3000
220  READ A:  IF A = 999 THEN  POKE  232,0:  POKE
233,144: GOTO 300
230  POKE R,A:R = R + 1: GOTO 220
300  GOSUB 940
400  CALL 18691
600 RD =  INT ( RND (1) * 4) + 1:A$(RD) = A$(RD) +
"7": XDRAW RD + 2 AT 32 + RD * 33,106 - 7 * 9
680  FOR T = 1 TO 4:LA =  LEN (A$(T)):  IF LA =  0
THEN 750
690 B$ = A$(T)
700   FOR  C = 1 TO LA:L$ =  MID$ (A$(T),C,1):L  =
VAL (L$)
710   IF L = 2 AND T = J THEN P = T:Q =  L:  GOSUB
1090: GOTO 730
720  XDRAW T + 2 AT 32 + T * 33,106 - L * 9
723 A =  VAL (B$) -  VAL ( LEFT$ (D$,LA - C + 1)):
IF A = 0 THEN B$ = "":MA = MA + .3: GOTO 745
724 B$ =  STR$ (A)
725  IF L = 1 THEN MA = MA + .3: GOTO 730
726 L = L - 1
727   XDRAW T + 2 AT 32 + T * 33,106 - L * 9:DE  =
4: GOSUB 2000
730   GOSUB  1200:  HTAB 10:KK =  INT (MA  +  .1):
PRINT KK;
740  NEXT
745 A$(T) = B$
750  NEXT T
760  IF GE < MA - 3 AND J = 0 THEN  GOSUB 840
800 AB =  LEN (A$(1) + A$(2) + A$(3) + A$(4)):  IF
AB > 12 THEN NS = 0
810  IF AB < NS / 3 + 1 THEN 600
820  GOTO 680
840  GOSUB 940:J = J + 1 -  INT (J / 5):  IF J = 5
THEN 1500
```

```
850  GOTO 940
900  GOSUB 940:J = J - 1 +  INT ((6 - J) / 5)
940  XDRAW 1 AT 32 + J * 33,99: RETURN
1090 CT = CT + 1:DE = 16:  GOSUB 2100:  XDRAW 1 AT
200 + 20 * CT,20:  IF CT = 3 THEN  XDRAW 1 AT 32 +
J * 33,99:DE = 17: GOSUB 2120: XDRAW 2 AT 28 + P *
33,106: GOTO 4060
1100 DE = 16:  GOSUB 2100: XDRAW P + 2 AT 32 + P *
33,106 - Q * 9: XDRAW 1 AT 32 + J * 33,99
1110  XDRAW 2 AT 28 + P * 33,106:DE =  17:  GOSUB
2120:  XDRAW  2 AT 28 + P * 33,106:J = 0:GE =  MA:
XDRAW 1 AT 32 + J * 33,99:  IF  LEN (B$) = 1  THEN
B$ = "": GOTO 1130
1120 B$ =  RIGHT$ (B$, LEN (B$) - 1)
1130  RETURN
1200  IF DD -  INT (DD / 10) * 10 = 0 THEN DD = DD
+ 1: GOSUB 4000
1205 KA =  PEEK ( - 16384): POKE  - 16368,0
1210  IF KA = 136 THEN DD = DD + 1: GOTO 900
1220  IF KA = 149 THEN DD = DD + 1: GOTO 840
1230  RETURN
1300                                         DATA
6,0,14,0,75,0,129,0,146,0,151,0,177,0,9,45,13,17,2
7,63,63,23,9,45,13,17,27,27,31,19,9,45,13,49,59,63
,63,19,9,45,13,17,59,63,63,19,9,45,13,17
1310                                         DATA
27,59,59,19,9,13,41,17,59,27,59,19,45,13,9,21,59,2
7,27,51,9,9,9,53,0,9,9,41,9,9,9,9,26,27,27,27,27,3
1,27,19,9,9,41,9,9,9,9,26,27,27
1320                                         DATA
59,63,63,59,55,45,45,45,45,45,9,9,62,63,63,63,63,6
3,59,55,9,9,9,9,9,9,9,2,0,45,9,9,17,63,63,63,55,45
,9,9,17,27,27,27,23,0,45,53,63,55
1330                                         DATA
0,41,9,9,9,21,63,31,27,59,55,45,45,45,45,53,63,31,
27,59,55,41,9,9,9,21,0,41,21,59,23,45,53,59,23,41,
21,59,23,41,17,0,999
1500  IF D THEN J = 4: GOTO 940
1505 MA = MA + 5
1510  GOSUB 940:DE = 5:  GOSUB 2000: GOSUB 940:J =
0:NS = NS + 1:GE = MA: GOTO 940
2000  FOR Z = DE TO 0 STEP  - 1:  POKE 777,Z * 3:
POKE 793,Z * 3: NEXT : RETURN
```

```
2010  RETURN
2100 DE = DE - 1: IF DE <  = 0 THEN  RETURN
2110  POKE 777,DE:  POKE 793,DE:  FOR Z = 1 TO 10:
NEXT : GOTO 2100
2120 DE = DE - 1: IF DE <  = 0 THEN  RETURN
2130  POKE 777,DE:  POKE 793,DE:  FOR Z = 1 TO 60:
NEXT : GOTO 2120
3000  HPLOT 0,115 TO 279,115
3010   HPLOT  8,92  TO  8,115:  HPLOT  200,92  TO
200,115: HPLOT 221,92 TO 221,115
3020  HPLOT 6,92 TO 22,80 TO 39,92:  HPLOT 6,91 TO
22,79 TO 39,91
3030   HPLOT  198,92 TO 210,80  TO  223,92:  HPLOT
198,91 TO 210,79 TO 223,91
3040  HPLOT 37,91 TO 37,115
3045   HPLOT 208,90 TO 212,90 TO 212,94 TO  208,94
TO 208,90
3050  RETURN
4000  IF D THEN D = 0: HCOLOR= 3: GOTO 4020
4010  HCOLOR= 0:D = 1
4020  HPLOT 199,97 TO 192,97 TO 192,114
4030  HPLOT 194,105: HCOLOR= 3: RETURN
4060  POKE 777,0:  POKE 793,0:  CALL 18700:  PRINT
"": PRINT "WANT TO PLAY AGAIN?";
4070  GET A$
4080  IF A$ = "Y" THEN  RUN
4090  IF A$ = "N" THEN  TEXT : HOME : END
5000  GOTO 4070
6000  POKE 18706,3:  POKE 18707,30:  POKE 18712,9:
POKE 18713,2:  POKE 18714,4:  POKE  18715,9:  POKE
19328,42
6010  FOR I = 768 TO 799: POKE I,0: NEXT
6020  POKE 774,16: POKE 790,16
6030  RETURN
```

(二)空戰驚魂 AIRFIGHT SHOOT OUT



一個第七期的遊戲，作者利用文字幕造成 ARCADE GAME 的效果。文字幕遊戲正是 BASIC 的優點，有時我們不必拚命向繪圖方面鑽研，巧妙地利用文字幕可得到意想不到的效果。況且亦是訓練編寫程式的結構，使更方便掌握編寫程式的初步技巧。

利用文字幕勢必運用很多變數和字串，所以第4行首先要加入 LOMEM 界限把變數移往20480，即第四頁後開始，以免侵佔第四頁的音樂資料。

第5行存入 MUSIC 和樂曲。

第40至55行輸入所需音樂演奏數值及產生背景樂曲。

原有程式聲音由600行和550行的副程式負責，這些地方必須改由 MOCKINGBOARD 產生聲音效果。

第700行至790行是 GAME OVER 後的聲音。

由於有背景音樂的關係，所以800行所有經由 APPLE SPEAKER 產生樂曲的部份都可以刪去，以免和 MOCKINGBOARD 衝突。

這個程式不長，値得初學編製遊戲的朋友作爲參考之用。

```
1  REM  ************************
2  REM  * AIRFIGHT SHOOT OUT   *
3  REM  ************************
4  LOMEM: 20480
5  D$ = CHR$ (13) + CHR$ (4): PRINT D$;"BLOAD
MUSIC": PRINT D$;"BLOAD MELODY,A$4000"
10  TEXT : HOME : NORMAL : SPEED= 255
20  GOSUB 860
30 SJ$ = "                                          "

40  POKE 18706,3: POKE 18707,4: POKE 18712,10:
POKE 18713,2: POKE 18714,4: POKE 18715,9: POKE
19328,42
45  POKE 774,16: POKE 790,16
50  FOR II = 768 TO 799: POKE II,0: NEXT
55  CALL 18691
60  POKE - 16368,0
70  PRINT
80 DS$ = CHR$ (92)
90  A$ = "                /" + DS$ + "
/" + DS$ + "      "
100  B$ = "     /" + DS$ + "     /  " + DS$ + "
/" + DS$ + "    /   " + DS$ + "      "
110  C$ = "  /   " + DS$ + "/" + DS$ + "/      /" +
DS$ + " /" + DS$ + "      /   " + DS$ + "/" + DS$ +
"/     " + DS$ + "      "
120  D$ = " /     " + DS$ + " " + DS$ + "   /  " +
DS$ + "  " + DS$ + "   /" + DS$ + "   / /      " +
DS$ + "   "
130  HOME : VTAB 2: HTAB 5: PRINT " @@@@ @   @
@@@   @@@  @@@@@           @     @   @ @   @ @   @
@              @@@   @@@@@  @    @  @    @    @
@  @   @ @   @ @   @   @              @@@@   @   @
@@@   @@@    @ "
140  PRINT : HTAB 11: PRINT " @@@  @   @  @@@@@
@    @  @   @   @                         @   @  @
@    @                       @    @  @     @    @
@@@   @@@     @"
150  PRINT : HTAB 10: PRINT "COMPUING-AGE
160  PRINT : HTAB 15: PRINT "PRESS A KEY"
170  VTAB 18: PRINT A$;B$;C$;D$
180  REM
190  REM
```

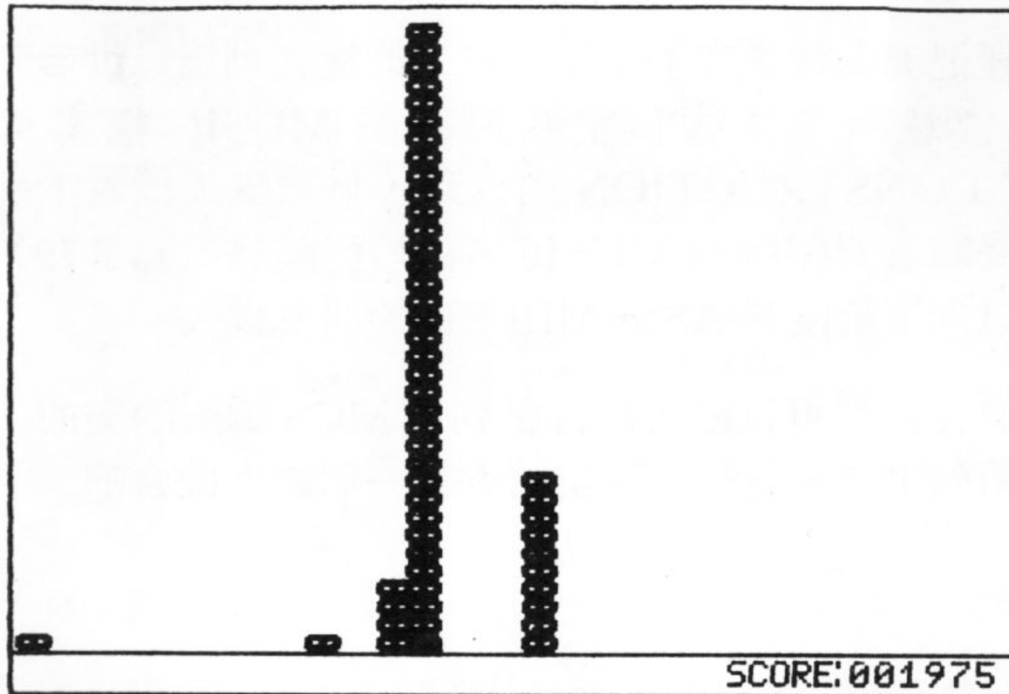
```
200  REM
210  IF  PEEK ( - 16384) < 128 THEN 210
220  POKE  - 16368,0
230              HOME              :SH$    =
"--------------------------------------"
240          VTAB          1:          PRINT        "
SCORE:00000"
250  VTAB 3: PRINT SH$: VTAB 22: PRINT SH$
260   VTAB 23:  PRINT "SHOOT-OUT      COMPUING-AGE
"
270 X = 12:OX = 12
280   IF  SO  < 6 AND  RND (1) * 10 >   =  8  THEN
GOSUB 490
290  IF SO = 0 THEN 370
300  FOR I = 1 TO SO:OX(I) = NX(I):OY(I) =  NY(I):
NEXT
310  FOR I = 1 TO SO:NY(I) = NY(I) - 2: IF NY(I) <
3 THEN  GOSUB 510
320  NEXT
330  FOR I = 1 TO SO
340 NX(I) = NX(I) +  INT ( RND (1) * 3 - 1)
350   IF   SCRN(  NY(I),2 * (NX(I) - 1))  +  16  *
SCRN( NY(I) - 1,2 * (NX(I) - 1) + 1) <  > 160 THEN
NX(I) = OX(I): GOTO 340
360  NEXT
370  VTAB 18: PRINT A$;B$;C$;D$
380  IF SO > 0 THEN  FOR I = 1 TO SO:  VTAB OX(I):
HTAB  OY(I):  PRINT " ":  VTAB NX(I):  HTAB NY(I):
INVERSE : PRINT "<": NORMAL : NEXT
390  IF N$ = "K" THEN  GOSUB 1060
400  IF N$ = "J" THEN  GOSUB 1140
410 A$ =  MID$ (A$,2) +  LEFT$ (A$,1)
420 B$ =  MID$ (B$,2) +  LEFT$ (B$,1)
430 C$ =  MID$ (C$,2) +  LEFT$ (C$,1)
440 D$ =  MID$ (D$,2) +  LEFT$ (D$,1)
450  IF SC < 0 THEN SC = 0
460   VTAB 1:  HTAB 37 - (SC > 10) - (SC > 100)  -
(SC > 1000) - (SC > 10000): PRINT SC
470  VTAB 1: HTAB 38: PRINT "  "
480  GOTO 280
490  SO  = SO + 1:NX(SO) =  INT ( RND (1) *  14  +
4):NY(SO) = 35
500  RETURN
```

```
510  VTAB OX(1): HTAB OY(1): PRINT " "
520  IF NX(1) = X THEN 690
530 SC = SC - 50
540  FOR T = 1 TO SO - 1:NX(T) = NX(T + 1):NY(T) =
NY(T  +  1):OX(T) = OX(T + 1):OY(T) = OY(T  +  1):
NEXT :SO = SO - 1: RETURN
550  POKE 776,12:  POKE 794,12:  FOR II = 1 TO 100
STEP 5: POKE 768,II: POKE 788,II: NEXT
555  POKE 776,0: POKE 794,0
560 SC = SC - 10
570  IF SO = 0 THEN  RETURN
580  FOR I = 1 TO SO: IF NX(I) = X THEN 620
590  NEXT
600  VTAB X:  HTAB 2: PRINT  MID$ (SH$,1,38): VTAB
X: HTAB 2: PRINT  MID$ (SJ$,1,38)
610  RETURN
620  VTAB X:  HTAB 2:  PRINT  MID$ (SH$,1,NY(I) -
2): VTAB X: HTAB 2: PRINT  MID$ (SJ$,1,NY(I) - 2)
630  FOR K = 16 TO 0
632  POKE 777,K: POKE 793,K
634  FOR J = 1 TO 200: NEXT
636  NEXT K
638   FOR J = 1 TO 200:  NEXT :  POKE 777,0:  POKE
793,0
640  VTAB NX(I): HTAB NY(I): PRINT "    "
650  FOR J = I TO SO - 1:NX(J) = NX(J + 1):NY(J) =
NY(J  + 1):OX(J) = OX(J + 1):OY(J) = OY(J  +  1):
NEXT J
660 SC = SC + 100
670 SO = SO - 1
680  RETURN
690  HOME
700   PRINT "          @@@    @     @     @  @@@@@
@      @    @  @   @@   @@  @                        @
@@@@@  @ @ @ @@@@                     @ @@@ @     @ @
@ @                         @@@   @     @ @    @ @@@@@"
710   PRINT  :  PRINT "          @@@   @    @  @@@@@
@@@@                    @    @ @    @  @        @     @
@    @ @    @ @@@@  @@@@                   @      @  @ @
@       @    @                      @@@     @   @@@@@  @
@"
720  POKE 776,12: POKE 794,12
```

```
725  FOR I = 250 TO 1 STEP  - 5:  POKE 768,I: POKE
788,250 - I: FOR J = 1 TO 50: NEXT J: NEXT I
730  PRINT : PRINT : PRINT "  WE REGRET THE SLIGHT
ACCIDENT YOU                    EXPERIENCED..."
740  FOR I = 250 TO 1 STEP  - 5:  POKE 768,I: POKE
788,250 - I: FOR J = 1 TO 100: NEXT J: NEXT I
750  PRINT : PRINT "  CONSOLATIONS    WILL BE SENT
TO YOUR             FAMILY."
760  FOR I = 200 TO 1 STEP  - 5:  POKE 768,I: POKE
788,200 - I: FOR J = 1 TO 20: NEXT J: NEXT I
770   PRINT :  PRINT "  YOUR SCORE   MIGHT  HAVE
BEEN  "(  INT  ( RND (1) * 10000) +  1):  PRINT  "
IT WAS   "SC
780  FOR I = 250 TO 1 STEP  - 5:  POKE 768,I: POKE
788,250 - I: FOR J = 1 TO 100: NEXT J: NEXT I
785  POKE 776,0: POKE 794,0
790  CALL 18700: CLEAR : GOTO 10
791  HTAB 15:  VTAB 23: PRINT "RESTART?";: GET A$:
IF A$ = "Y" THEN  RUN
792  IF A$ <  > "N" THEN 791
793  HOME : NEW
800  REM
810  REM
820  REM
830  REM
840  REM
850  REM
860  HOME : VTAB 1: HTAB 15: PRINT "SHOOT - OUT"
870  PRINT : PRINT "YOU ARE THE PILOT OF A SMALL"
880  PRINT : PRINT "RECONAISSANCECRAFT FLYING OVER
THE
890  PRINT : PRINT "MOUNTAINOUS MOON OF LUF IN THE
OBSCURE
900   PRINT :  PRINT "KULY  GALAXY.SUDDENLY,HOSILE
ORKANS FLY
910   PRINT :  PRINT "ONTO THE SCENE AND  ADOPT  A
HYPNOTIC
920   PRINT  :  PRINT "DANCE TO TRY  TO  HYPNOTISE
YOU.YOUR ONLY
930  PRINT "CHANCE IS TO SHOOT THEM BEFORE THEY
940   PRINT :  PRINT "REACH THE OTHER SIDE OF  THE
SCREEN.THIS
950  PRINT "COSTS YOU 50 POINTS,A SHOT TEN.IF YOU
```

```
960  PRINT :  PRINT "SCORE A DIRECT HIT ON ONE OF
THE SHIPS ..
970  PRINT :  PRINT "          PLEASE HIT A KEY ";:
GET A$
980  PRINT : PRINT : HOME
990  PRINT : PRINT "YOU SCORE 100 POINTS.USE<-AND-
>TO
1000  PRINT :  PRINT "MOVE YOUR SHIP UP AND  DOWN
AND SPACE
1010  PRINT : PRINT "BAR TO FIRE A PROTON TORPEDO.
1020  PRINT : PRINT "      GOOD LUCK!"
1030  PRINT  :  PRINT " PRESS  'K'  FOR  KEYBOARD
CONTROL     OR         'J' FOR JOYSTICKS ";:  GET
N$
1040  IF N$ <  > "J" AND N$ <  > "K" THEN 860
1050  RETURN
1060  IF  PEEK ( - 16384) = 136 THEN X = X - 1
1070  IF  PEEK ( - 16384) = 149 THEN X = X + 1
1080  IF  SCRN( 0,2 * (X - 1)) + 16 *  SCRN( 0,2 *
(X - 1) + 1) <  > 160 AND  SCRN( 0,2 * (X - 1))  +
16 *  SCRN( 0,2 * (X - 1) + 1) <  > 190 THEN 690
1090  HTAB 1: VTAB OX: PRINT " "
1100  VTAB X: HTAB 1: PRINT ">"
1110 OX = X
1120  IF  PEEK ( - 16384) = 160 THEN  GOSUB 550
1125  POKE  - 16368,0
1130  RETURN
1140  HTAB 1: VTAB X: PRINT " "
1150  IF  PDL (1) < 20 THEN X = X - 1
1160  IF  PDL (1) > 235 THEN X = X + 1
1170  HTAB 1: VTAB OX: PRINT " "
1180  IF  SCRN( 0,2 * (X - 1)) + 16 *  SCRN( 0,2 *
(X  - 1) + 1) <  > 160 AND  SCRN( 0,2 * (X - 1)) +
16 *  SCRN( 0,2 * (X - 1) + 1) <  > 190 THEN 690
1190  VTAB X: HTAB 1: PRINT ">"
1200 OX = X
1210  IF  PEEK ( - 16287) > 127 THEN  GOSUB 550
1220  RETURN
```

(三)太空侵略者封閉戰 CLOSING ALIEN



電腦時代第16期難得的作品。和其他太空侵略者遊戲不同，遊戲運行並不覺得緩慢，而且甚有趣味性。作者放棄一味只顧盲目射擊的玩法，代之而把太空侵略者控制他們降落位置，並將他們疊積起來得分，一個適合 BASIC 速度巧妙構思的遊戲。

該程式應用第4頁的高解像第二頁作爲繪圖，所以音樂部份不能再放在同一頁。必須修改 MUSIC 原始碼，經組合後放在＄7000開始的第7頁。（如何修改可參閱同期的關於背景音樂專欄或見附錄）。

第1410行便是把圖形表，修改後的 MUSIC 和樂曲資料存放在適當記憶體內。

第670行是激光發射的聲音效果，而第650行是爆炸效果。

至於如何修改，相信讀者已經有一個概念了。

這裡把加上背景音樂和聲音效果的程式列出以供參考之用。請留意高空擲物和空戰驚魂的 MUSIC 檔案可在 MUSIC CONSTRUCTION 內找到，無須鍵入以省工夫。而太空侵略者封閉戰的 MUSIC，由於已經修改爲＄7900開始，所以列出給沒有 ASSEMBLER 的朋友鍵入。

所有程式都可以輸入你自己利用 MCS 編製的樂曲，因此亦無須列出音樂資料，就留給大家一個創作機會吧。

```
10  REM    **********************
20  REM    *     CLOSING ALIEN    *
30  REM    *   BY COMPUHOUSE      *
40  REM    * COPYRIGHT (C) 1985 *
50  REM    *  BY COMPUTING AGE  *
60  REM    *                      *
70  REM    **********************
80  LOMEM: 25600
90  ONERR  GOTO 1560
100  GOSUB 1410
110  GOSUB 1430: GOTO 130: REM     INITIALIZE
120  CLEAR : GOSUB 1410
130 SC$ = "000000":SC = 0
135  POKE 31616,42:  POKE 31000,13:  POKE 31001,2:
POKE 31002,4: POKE 31003,8
136  FOR II = 768 TO 799:  POKE II,0:  NEXT : CALL
30979
140   DIM  R(14),SC(13):  FOR I = 0  TO  13:  READ
SC(I):          NEXT          :          DATA
100,50,25,10,5,5,5,5,5,5,10,25,50,100
150  HGR2 : SCALE= 1: ROT= 0: HCOLOR= 3:X = 130
160   HPLOT  0,0 TO 279,0 TO 279,180 TO  0,180  TO
0,0: HPLOT 279,180 TO 279,191 TO 0,191 TO 0,180
180  DRAW 14 AT 203,182: GOSUB 760
190 MI = 12:MA = 252:Y1 = 9
200  HCOLOR= 3:Z =  PEEK ( - 16384): IF Z = 136 OR
Z  = 149 THEN  POKE  - 16368,0:M =  ABS (X - X1):X
=  INT (X + (5 * (M <  = 20) + .5 * (M > 20) *  M)
* ((Z = 149) - (Z = 136)))
210  IF MA - MI < 20 THEN 830
220   IF SC - RE >  = 10000 THEN RE = RE +  10000:
FOR I = 0 TO 13:R(I) = 0: NEXT : GOTO 150
230   IF Y1 = 9 AND  INT (( RND (1) * 10) / 3) = (
INT ( RND (0) * 10)) / 3 THEN 990
240  IF X < MI THEN X = MI: GOTO 260
250  IF X > (MA) THEN X = MA
260  XDRAW 11 AT X,179
270  IF Y1 > 9 THEN 290
280 R =  INT ( RND (1) * 14):X1 = (R * 20) + 2: IF
X1 < MI OR X1 > (MA) THEN 280
290  GOSUB 380: REM   CHECK POSITION
300  DRAW 12 AT X1,Y1
310 Z =  PEEK ( - 16384): IF Z = 160 THEN  POKE  -
16368,0: GOSUB 470
```

```
320  GOSUB 410
330  HCOLOR= 0
340  IF Y1 < = 179 THEN  DRAW 12 AT X1,Y1:Y1 = Y1
+ 5
350  IF Y1 > 169 THEN  GOSUB 860
360  IF Y1 = 179 OR Y1 = 179 - (R(R) * 5) THEN  Y1
= 9:R(R) = R(R) + 1: GOSUB 710: GOSUB 420
370  GOTO 200
380  REM  :CHECK FOR TOP OF COLUMN POSITION
390  IF Y1 = 179 - (R(R) * 5) - 5 THEN  HCOLOR= 3:
XDRAW  12  AT  X1,Y1 + 5:Y1 = 9:R(R) = R(R)  +  1:
GOSUB 410: GOSUB 710: GOSUB 420: POP : GOTO 200
400  RETURN
410  XDRAW 11 AT X,179: RETURN
420  REM  :SET RANGE OF GUN
430  IF X1 = MI OR X1 = (MA) THEN  RETURN
440   IF  X1 < X AND X1 > MI THEN MI =  X1  +  10:
RETURN
450  IF X1 > X AND X1 < (MA) THEN MA = X1 - 10
460  RETURN
470  REM  :FIRE SHOT
480   HPLOT  X  + 5,169 TO X +  5,Y1:  GOSUB  680:
HCOLOR= 0: HPLOT X + 5,169 TO X + 5,Y1: HCOLOR= 3
490   IF X = X1 OR X = X1 + 1 OR X = X1  - 1  THEN
560
500   IF X = X1 + 2 OR X = X1 + 3 OR X = X1 + 4 OR
X = X1 + 5 THEN  GOSUB 600:X1 = X1 - 20:R = R - 1:
GOTO 530
510  IF X = X1 - 2 OR X = X1 - 3 OR X = X1 - 4  OR
X = X1 - 5 THEN  GOSUB 620:X1 = X1 + 20:R = R + 1:
GOTO 530
520  RETURN
530  IF X1 < 2 THEN X1 = 2: RETURN
540  IF X1 > 262 THEN X1 = 262
550  RETURN
560  REM  :RANDOM MOVE FOR CENTER STRIKE
570 RM =  INT ( RND (1) * 2) + 1: IF RM / 2 =  INT
(RM / 2) THEN 590
580  GOSUB 620:X1 = X1 + 20:R = R + 1:  GOSUB 380:
GOTO 530
590  GOSUB 600:X1 = X1 - 20:R = R - 1:  GOSUB 380:
GOTO 530
600 TX = X1 - 20:TY = Y1 + 5: IF 179 - (R(R - 1) *
5) <  = TY THEN  POP : POP : GOTO 320
```

```
610  HCOLOR= 0: DRAW 12 AT X1,Y1: HCOLOR= 3:
RETURN
620  IF R = 13 THEN  POP : POP : GOTO 320
630 TX = X1 + 20:TY = Y1 + 5: IF 179 - (R(R + 1) *
5) < = TY THEN  POP : POP : GOTO 320
640  GOTO 610
650  REM   : SOUND OF EXPLOSION
660 FF = 17
662 FF = FF - 1: IF FF < = 0 THEN 668
664  POKE 777,FF: POKE 793,FF: FOR JJ = 1 TO 50:
NEXT
666  GOTO 662
668  FOR JJ = 1 TO 100: NEXT : RETURN
670  REM  :SOUND OF SHOT
680  POKE 776,12: POKE 794,12
690  FOR FF = 1 TO 240 STEP 10: POKE 768,FF: POKE
788,FF: NEXT
700  POKE 776,0: POKE 794,0: RETURN
710  REM  :UPDATE SCORE
720  HCOLOR= 0: FOR J = 1 TO 6:C = ASC ( MID$
(SC$,J,1))      - 47:     ON     J     GOSUB
770,780,790,800,810,820: NEXT : HCOLOR= 3
730  IF FL THEN  RETURN
740 SC = SC + SC(R):SC$ =  STR$ (SC)
750  IF  LEN (SC$) < 6 THEN SC$ = "0" + SC$: GOTO
750
760  FOR J = 1 TO 6:C = ASC ( MID$ (SC$,J,1)) -
47: ON  J GOSUB 770,780,790,800,810,820: NEXT :
RETURN
770  DRAW C AT 233,188: RETURN
780  DRAW C AT 240,188: RETURN
790  DRAW C AT 247,188: RETURN
800  DRAW C AT 254,188: RETURN
810  DRAW C AT 261,188: RETURN
820  DRAW C AT 268,188: RETURN
830  DRAW 11 AT MI,179
840  FOR I = 9 TO 179 STEP 5: DRAW 12 AT MI,I:
POKE  - 16336, PEEK ( - 16336): NEXT
850 X = MI: GOTO 880
860  IF X = X1 OR X = X1 + 1 OR X = X1 + 2 OR X =
X1  + 3 OR X = X1 + 4 OR X = X1 + 5 OR X = X1  - 1
OR  X = X1 - 2 OR X = X1 - 3 OR X = X1 - 4 OR X  =
X1 - 5 THEN 880
870  RETURN
```

```
880  HCOLOR= 3:  GOSUB 650: FOR I = 1 TO 5: SCALE=
I: DRAW 11 AT X,179: NEXT
890  HCOLOR= 0:  GOSUB 650: FOR I = 1 TO 5: SCALE=
I: DRAW 11 AT X,179: NEXT : SCALE= 1
900  FOR I = 1 TO 1000: NEXT : TEXT : HOME
910  IF SC < HS THEN  POKE 777,0: POKE 793,0: CALL
30988: GOTO 960
920   VTAB  12:  FLASH :  PRINT  "NEW  HIGH  SCORE
:"SC:HS = SC: NORMAL : PRINT
925  POKE 770,0: POKE 793,0: CALL 30988
930  INPUT "PLEASE ENTER YOUR NAME :";PL$:  IF PL$
= "" THEN 930
940  PRINT  CHR$ (4)"OPEN HIGH.SCORE": PRINT  CHR$
(4)"WRITE HIGH.SCORE":  PRINT HS: PRINT PL$: PRINT
CHR$ (4)"CLOSE"
950  GOTO 970
960   VTAB 12:  PRINT "HIGH SCORE IS ";:  FLASH  :
PRINT HS;: NORMAL : PRINT " BY "PL$"."
970  VTAB 22:  INPUT "PLAY AGAIN ?";YN$: IF  LEFT$
(YN$,1) = "Y" THEN  FOR I = 0 TO 14:R(I) = 0: NEXT
:SC = 0:SC$ = "000000": GOTO 120
980  HOME :  FOR I = 38 TO 1 STEP  - 1: GOSUB 650:
VTAB 24: HTAB I: PRINT  CHR$ (93)" ";: NEXT : VTAB
23: END
990  REM  :BOMB SEQUENCE
1000 B =  INT ( RND (1) * 3 + 1)
1010  S =  INT ( RND (1) * 5) * 10:S = Y1 + S:Y1 =
S
1020  HCOLOR= 3:Z =  PEEK ( - 16384):  IF Z =  136
OR  Z  = 149 THEN  POKE  - 16368,0:M =  ABS  (X  -
X1):X =  INT (X + (5 * (M <  = 20) + .5 * (M > 20)
* M) * ((Z = 149) - (Z = 136)))
1030  IF X < MI THEN X = MI
1040  IF X > (MA) THEN X = MA
1050  XDRAW 11 AT X,179
1060  IF Y1 > S THEN 1090
1070  R =  INT ( RND (1) * 14):X1 = (R * 20) +  2:
IF X1 < MI OR X1 >  (MA) THEN 1070
1080  FOR I = 1 TO B:B(I) = X1:X1 = X1 + 5: NEXT
1090  FOR I = 1 TO B: DRAW 13 AT B(I),Y1: NEXT
1100  Z =  PEEK ( - 16384):  IF Z = 160 THEN  POKE
- 16368,0: GOSUB 1190
1110  HCOLOR= 0
```

```
1120  FOR I = 1 TO B: DRAW 13 AT B(I),Y1: NEXT
1130  DRAW 11 AT X,179:  FOR L = 168 TO 170: HPLOT
X + 3,L TO X + 6,L: NEXT : GOSUB 410:Y1 = Y1 + 10
1140   FOR  I = 1 TO B:  IF Y1 = 179  THEN  GOSUB
1360: IF B = I THEN BG = 1:Y1 = 9: GOSUB 410
1150  NEXT
1160  IF EG = 1 THEN EG = 0:BG = 0: GOTO 880
1170  IF BG = 1 THEN BG = 0: GOTO 200
1180  GOSUB 410: GOTO 1020
1190  REM  :SHOOT AT BOMBS
1200  HPLOT X + 5,169 TO X + 5,Y1 + 1:  GOSUB 680:
HCOLOR=  0:  HPLOT  X  + 5,169 TO X +  5,Y1  +  1:
HCOLOR= 3
1210   FOR J = 1 TO B:  IF X > B(J) - 2.5 AND X  <
B(J) + 2.5 THEN I = J:J = B:EX = 1
1220  NEXT
1230  IF EX = 1 THEN EX = 0: GOTO 1250
1240  RETURN
1250  HCOLOR= 3: GOSUB 650: FOR K = 1 TO 3: SCALE=
K: XDRAW 13 AT B(I),Y1: NEXT
1260  HCOLOR= 0: GOSUB 650: FOR K = 2 TO 3: SCALE=
K: XDRAW 13 AT B(I),Y1: NEXT : SCALE= 1
1270 FL = 1: GOSUB 710:FL = 0
1280 SC = SC + 150:SC$ =  STR$ (SC)
1290  IF  LEN (SC$) < 6 THEN SC$ = "0" + SC$: GOTO
1290
1300  GOSUB 760
1310  IF I = B AND B = 1 THEN Y1 = 9:  HCOLOR=  0:
DRAW  11 AT X,179:  FOR L = 168 TO 170:  HPLOT X +
3,L TO X + 6,L: NEXT : POP : GOTO 200
1320  IF I = B THEN B = B - 1: RETURN
1330  IF I = 1 AND B = 2 THEN B(I) = B(I + 1):B  =
1: RETURN
1340   IF I = 1 AND B = 3 THEN B(I) = B(I + 1):B(I
+ 1) = B(I + 2):B = 2: RETURN
1350  IF I = 2 AND B = 3 THEN B(I) = B(I + 1):B  =
2: RETURN
1360   IF  X > B(I) - 2.5 AND X < B(I) + 2.5  THEN
HCOLOR= 3:  GOSUB 650:  FOR K = 1 TO 3:  SCALE= K:
XDRAW 13 AT B(I),179: NEXT
1370   IF X > B(I) - 2.5 AND X < B(I) +  2.5  THEN
HCOLOR= 0:  GOSUB 650:  FOR K = 1 TO 3:  SCALE= K:
XDRAW  13 AT B(I),179:  NEXT :  HCOLOR= 3:  SCALE=
```

```
1:EG = 1: RETURN
1380  HCOLOR= 3: GOSUB 650: FOR K = 1 TO 3: SCALE=
K:  XDRAW 13 AT B(I),179:  NEXT : HCOLOR= 0: GOSUB
650:  FOR  K  = 1 TO 3:  SCALE=  K:  XDRAW  13  AT
B(I),179: NEXT : HCOLOR= 3: SCALE= 1
1390  IF B(I) / 20 <  >  INT (B(I) / 20) THEN B(I)
= B(I) - 1: GOTO 1390
1400 X1 = B(I): GOSUB 420: RETURN
1410  PRINT  CHR$ (4)"VERIFY  HIGH.SCORE":  PRINT
CHR$  (4)"OPEN HIGH.SCORE":  PRINT  CHR$  (4)"READ
HIGH.SCORE": INPUT HS,PL$: PRINT  CHR$ (4)"CLOSE"
1420  RETURN
1430  TEXT : HOME
1440  VTAB 12:  HTAB 13:  INVERSE : PRINT "CLOSING
ALIEN ":  PRINT :  HTAB 12: PRINT "BY:  COMPUHOUSE
":  PRINT : PRINT "** COPYRIGHT 1985 BY  COMPUTING
AGE  **": NORMAL
1450  IF HS > 0 THEN  PRINT :  PRINT "PRESENT HIGH
SCORE IS "HS" BY "PL$
1460 D$ =  CHR$ (13) +  CHR$ (4):  PRINT D$;"BLOAD
ALIEN SHAPE":  PRINT D$;"BLOAD ALIEN MUSIC": PRINT
D$;"BLOAD ALIEN MELODY,A$7000"
1470  POKE 232,0: POKE 233,96
1480  VTAB  22:  INPUT "DO YOU NEED  INSTRUCTIONS
?";YN$: IF  LEFT$ (YN$,1) = "N" THEN  RETURN
1490  HOME :  HTAB 14:  INVERSE :  PRINT  "CLOSING
ALIEN ": NORMAL : PRINT
1500  VTAB 12:  INVERSE :  PRINT "PLEASE REFER TO
THE MAGAZINE": NORMAL
1540  VTAB 21: PRINT "PRESS ANY KEY TO CONTINUE";:
GET YN$
1550  RETURN
1560  REM   : ERROR ROUTINE
1570  ER  =  PEEK (222):EL =  PEEK (218)  +  PEEK
(219) * 256: POKE 216,0
1580  IF  ER  =  6  THEN  PRINT  CHR$  (4)"OPEN
HIGH.SCORE":  PRINT  CHR$ (4)"WRITE  HIGH.SCORE":
PRINT 0: PRINT : PRINT  CHR$ (4)"CLOSE": RUN
1590  IF ER = 255 THEN  TEXT : GOTO 980
1600  TEXT :  HOME :  VTAB 12:  PRINT "ERROR #"ER"
HAS OCCURRED IN LINE "EL
1610  PRINT  :  PRINT  "PLEASE  REFER  TO  YOUR
APPLESOFT MANUAL."
```

ALIEN MUSIC,A$7900,L$2ED

```
7900- 4C B1 79 4C 7C 7B 4C C8
7908- 7A 4C 3D 7B 4C D8 7B 4C
7910- EB 7B 03 03 FE 6F 7E 74
7918- 0A 00 06 0A 00 00 00 00
7920- 00 00 00 00 01 01 FF FF
7928- FF FF FF FF 00 00 00 00
7930- 00 1E 1F 20 22 24 26 29
7938- 2C 2E 30 33 36 3A 3D 41
7940- 45 49 4D 52 56 5C 61 67
7948- 6D 73 7A 81 89 91 9A A3
7950- AD B7 C2 CE DA E7 F4 03
7958- 12 23 34 46 5A 6E 84 9B
7960- B3 CD E9 06 25 45 68 8C
7968- B3 DC 08 36 67 9B D2 01
7970- 01 00 00 00 00 00 00 00
7978- 00 00 00 00 00 00 00 00
7980- 00 00 00 00 00 00 00 00
7988- 00 00 00 00 00 00 00 00
7990- 00 00 00 00 00 00 00 01
7998- 01 01 01 01 01 01 01 01
79A0- 01 01 01 02 02 02 02 02
79A8- 02 02 03 03 03 03 03 00
79B0- 00 8A 48 98 48 A9 C0 8D
79B8- 0D C4 EE 1D 79 EE 1C 79
79C0- AD 1D 79 CD 13 79 D0 08
79C8- 20 A0 7B A9 00 8D 1D 79
79D0- AD 1C 79 CD 12 79 F0 03
79D8- 4C 6F 7A A9 00 8D 1C 79
79E0- A2 00 8E 21 79 8A 0A 8D
79E8- 22 79 DE 24 79 BD 24 79
79F0- D0 6F 20 1E 7B AE 22 79
79F8- B5 04 18 69 02 95 04 90
7A00- 02 F6 05 A1 04 4A 8D 1F
```

```
7A08-  79 F6 04 A1 04 D6 04 8D
7A10-  20 79 0D 1F 79 D0 06 20
7A18-  3D 7B 4C 6C 7A AE 21 79
7A20-  20 76 7A AD 21 79 9D 26
7A28-  79 8E 23 79 AE 1F 79 BD
7A30-  31 79 8D 2D 79 BD 71 79
7A38-  8D 2E 79 AD 18 79 8D 2C
7A40-  79 AE 23 79 20 9A 7A AE
7A48-  21 79 AD 20 79 29 40 9D
7A50-  2F 79 AD 20 79 29 3F 9D
7A58-  24 79 AD 20 79 29 80 D0
7A60-  94 AE 21 79 E8 E0 02 F0
7A68-  03 4C E2 79 20 C8 7A 68
7A70-  A8 68 AA A5 45 40 E0 00
7A78-  D0 10 AE 19 79 BD 26 79
7A80-  30 17 E8 EC 1A 79 D0 F5
7A88-  CA 60 AE 1A 79 CA BD 26
7A90-  79 30 06 CA EC 19 79 D0
7A98-  F5 60 8A 48 98 48 A9 00
7AA0-  E0 03 90 05 CA CA CA A9
7AA8-  10 85 D6 8A 0A A8 AD 2D
7AB0-  79 91 D6 C8 AD 2E 79 91
7AB8-  D6 8A 18 69 08 A8 AD 2C
7AC0-  79 91 D6 68 A8 68 AA 60
7AC8-  98 48 A9 FF 8D 03 C4 8D
7AD0-  83 C4 A9 07 8D 02 C4 8D
7AD8-  82 C4 A0 00 8C 01 C4 A9
7AE0-  07 8D 00 C4 A9 04 8D 00
7AE8-  C4 B9 00 03 8D 01 C4 A9
7AF0-  06 8D 00 C4 A9 04 8D 00
7AF8-  C4 8C 81 C4 A9 07 8D 80
7B00-  C4 A9 04 8D 80 C4 B9 10
7B08-  03 8D 81 C4 A9 06 8D 80
7B10-  C4 A9 04 8D 80 C4 C8 C0
7B18-  0F D0 C1 68 A8 60 AE 19
7B20-  79 BD 26 79 CD 21 79 D0
```

```
7B28- 0D A9 FF 9D 26 79 A9 00
7B30- 8D 2C 79 20 9A 7A E8 EC
7B38- 1A 79 D0 E5 60 AD 14 79
7B40- 85 04 AD 15 79 85 05 AD
7B48- 16 79 85 06 AD 17 79 85
7B50- 07 A9 00 8D FE 03 A9 79
7B58- 8D FF 03 A9 00 85 D6 A9
7B60- 03 85 D7 A9 01 8D 24 79
7B68- 8D 25 79 A9 00 8D 21 79
7B70- 20 1E 7B A9 01 8D 21 79
7B78- 20 1E 7B 60 20 3D 7B A9
7B80- F8 8D 07 03 8D 17 03 A9
7B88- 40 8D 0B C4 A9 C0 8D 0D
7B90- C4 8D 0E C4 A9 FF 8D 04
7B98- C4 A9 40 8D 05 C4 58 60
7BA0- A2 00 BD 26 79 C9 02 B0
7BA8- 11 A8 B9 2F 79 D0 0B BD
7BB0- 08 03 CD 1B 79 F0 03 DE
7BB8- 08 03 BD 29 79 C9 02 B0
7BC0- 11 A8 B9 2F 79 D0 0B BD
7BC8- 18 03 CD 1B 79 F0 03 DE
7BD0- 18 03 E8 E0 03 D0 CB 60
7BD8- 78 A9 00 8D 21 79 20 1E
7BE0- 7B EE 21 79 20 1E 7B 20
7BE8- C8 7A 60 58 60 83
```

ALIEN SHAPE ,A$6000,L$21E

```
6000-  12 00 26 00 35 00 3E 00
6008-  4A 00 56 00 61 00 6D 00
6010-  79 00 81 00 8E 00 99 00
6018-  B2 00 CB 00 D6 00 15 01
6020-  74 01 D2 01 05 02 29 2D
6028-  20 24 24 3B 3F 32 36 2E
6030-  28 28 20 00 00 2D 2D DC
6038-  24 24 BC 17 04 00 2D 2D
6040-  DC 1B 64 2D 05 20 1C 3F
6048-  27 00 A8 2D 05 20 1C 67
6050-  21 1C 3F 17 04 00 49 24
6058-  05 38 3F 27 0C 0C 35 26
6060-  00 A8 2D 05 20 1C 3F 27
6068-  24 2D 2D 04 00 29 2D 20
6070-  1C 3F D6 24 0C 0C 0C 04
6078-  00 21 64 0C 0C 3C 3F 27
6080-  00 29 2D 20 1C 67 21 1C
6088-  3F 17 AE 1E 26 00 29 28
6090-  28 20 E4 3F 17 76 2D 04
6098-  00 2D 2D 2D 2D 25 3F 3F
60A0-  3F 3F 67 2D 2D 2D E5 3F
60A8-  3F 67 2D E5 27 25 27 25
60B0-  27 00 29 2D 2D 2D 05 38
60B8-  3F 3F 3F 3F 2C 4D 6D 29
60C0-  3C 3F 3F 3F 3F 0C 2D 2D
60C8-  2D 25 00 49 29 05 38 3F
60D0-  2C 2D 1C 27 25 00 3F BF
60D8-  76 2D 15 F6 3F 67 49 51
60E0-  29 2D F8 1B 24 24 0C 2D
60E8-  15 0D 36 36 0E 2D 05 20
60F0-  24 E4 3F 4C 49 32 36 36
60F8-  6E 09 1C 1C 07 28 2D 20
```

```
6100-  1C  3F  4C  49  32  36  36  2E
6108-  2D  25  C0  3F  04  80  28  2D
6110-  0D  B6  12  26  00  07  38  3F
6118-  32  36  36  29  2D  20  01  08
6120-  18  08  30  36  36  2E  2D  2D
6128-  21  24  24  21  29  2A  32  3E
6130-  3F  0D  0A  2E  08  18  08  18
6138-  08  30  36  36  0E  2D  05  20
6140-  24  24  0D  09  09  3F  37  33
6148-  0E  2D  15  36  3B  3F  0F  09
6150-  09  09  18  08  18  08  18  08
6158-  2D  2D  1D  1B  32  36  36  0D
6160-  18  08  01  08  29  2D  20  04
6168-  3B  3F  36  36  36  0D  18  08
6170-  15  15  05  00  36  36  0E  2D
6178-  05  20  24  24  3B  3F  09  09
6180-  09  12  0A  2D  05  20  1C  3F
6188-  37  36  36  2E  09  09  21  24
6190-  24  2C  09  31  36  3F  37  09
6198-  31  2E  01  20  24  24  29  2D
61A0-  32  36  36  3B  3F  09  09  09
61A8-  24  24  24  2D  2D  32  1E  3F
61B0-  0F  11  09  36  3B  3F  09  09
61B8-  09  2D  1D  03  20  24  24  2F
61C0-  29  12  0A  12  22  24  24  21
61C8-  31  31  31  3E  3F  0E  09  2E
61D0-  00  05  36  36  2D  05  E0  27
61D8-  28  15  45  76  2D  20  95  DA
61E0-  37  6D  29  3C  04  20  35  0D
61E8-  08  36  36  4D  1C  07  28  05
61F0-  F8  28  4D  36  76  2D  20  24
61F8-  0D  36  36  2D  05  20  FC  28
6200-  4D  76  36  04  00  31  1E  36
6208-  4D  09  2D  20  BF  23  64  2D
6210-  0D  36  76  2D  20  24  0D  76
6218-  36  25  2C  20  04  00  38
```

後記

本書至此可算告一段落了，相信各位經常編寫 BASIC 的讀者都能充份利用 MOCKINGBOARD 的效果容納在程式內。雖然 BASIC 所能應用有限，如果需要更佳的音响效果，當然非機械語言的幫助不可。畢竟 MOCKING-BOARD 本身是一個輸入輸出硬件，處理硬件不能不應用直接運行的低階層語言。不過 BASIC 所編寫的程式亦因適當運行達到某些效果。

隨書附有 3 隻磁碟，內裡存有所有本書提及的示範程式，亦可幫助大家免去鍵入之苦。

有機會再更加深入討論 MOCKINGBOARD 的其他應用技巧。





電腦時代出版社
COMPUTING AGE PUBLISHER
九龍中央郵政信箱 71193 號
電話：3 — 312007

本證是用來購買本書內所刊登過之程式列表之磁碟版本，共有三張磁碟：

☐ (1)DATA DISK（2 SIDE：SIDE A FOR MUSI-C LIST CREATOR ; SIDE B FOR LONG MUSIC CREATOR）（第四章）
☐ (2)MUSIC LIST CREATOR／LONG MUSIC CRE-ATOR（第四章）
☐ (3)DEMO GAMES（第五章）

磁碟每張 25 元。大家必須剪下此證（影印本無效）寄來電腦時代出版社。

上列各項軟件將會附同有一張購買證用來購買麥法基先生專爲本書讀者而寫的 MOCKING BOARD 版遊戲及補充資料。所有這些遊戲及補充資料都是從未發表過，亦不會公開發表。

請將上列軟件寄給本人，（請在☐內加✓）
本人姓名：________________ 電話：____________
地址：____________________________

MOCKING BOARD 專輯軟件	$	
郵費及處理費	$	4
掛號郵費	$	4
總共	$	

支票號碼＃__________ 銀行＃__________

please allow 4 to 6 weeks for delivery
＊售價如有調整，恕不另行通告

請將此頁從書上剪下及寄來電腦時代出版社（必須正本，影印本無效）。每份補充資料將會附同下一份補充資料的索取券。

電腦時代出版社出版
電腦技術應用叢書

(1) 128 K RAM 咭應用專輯
(2) GPLE 使用方法深入研究
(3) MOCKING BOARD 技術應用
(4) 電腦時代 1-12 期精裝合訂本

書名：MOCKING BOARD 應用技術
作者：麥法基
出版：電腦時代出版社
發行：電腦時代讀者服務部
旺角上海街438號同珍商業中心
1104室 3-312007
出版日期：一九八五年七月（初版）
定價 價：每本港幣三十元
出版書號：85003